IPv6 Configuration Guide, Cisco IOS Release 15.2S

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

Start Here: Cisco IOS Software Release Specifics for IPv6 Features

The IPv6 for Cisco IOS software feature documentation provides implementation and command reference information for IPv6 features supported in the Cisco IOS software. This Start Here document details only the Cisco IOS software release specifics for IPv6 features. Not all IPv6 features may be supported in your Cisco IOS software release. We strongly recommend that you read this entire document before reading the other IPv6 for Cisco IOS software feature documentation.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Cisco IOS Software Platform Dependencies and Restrictions

See the table below to determine which IPv6 features are supported in each release of the Cisco IOS software trains.

Note For information about IPv6 features in Cisco IOS XE software releases, see "Start Here: Cisco IOS XE Software Release Specifics for IPv6 Features."

- IPv6 was introduced on the 12.0(21)ST Cisco IOS software release train, which was merged with the 12.0S Cisco IOS software release train starting at Cisco IOS Release 12.0(22)S. The 12.0S Cisco IOS software release train provides IPv6 support on Cisco 12000 series Internet routers and Cisco 10720 Internet routers only.

- The 12.2S Cisco IOS release train comprises a family of release trains, each supporting different platforms as follows:
  - The 12.2SB Cisco IOS release train comprises the Cisco 10000, 7304, 7301, and 7200 series. As of Cisco IOS Release 12.2(33)SB, the Cisco 7200 and 7301 series are not supported on the 12.2SB release train.

The 12.2SE Cisco IOS release train consists of the Cisco Catalyst 3560, 3750, 3560E, 3750E series, and the Cisco Catalyst 3750 Metro series.

- The 12.2SG Cisco IOS release train consists of the Cisco Catalyst 4500 and Cisco Catalyst 4900 series.
- The 12.2SR Cisco IOS release train consists of the Cisco 7600 and 7200 series routers.
- The 12.2SX Cisco IOS release train consists of the Cisco Catalyst 6500. Before the 12.2SR Cisco IOS release train, the 12.2SX release train also included the Cisco 7600 series.
- The 15.0M, 15.1T, and 15.2T Cisco IOS release trains are a continuation of the 12.2, 12.3, and 12.4 Cisco IOS release trains.
- IPv6 is also supported in some special software release trains.

Cisco IOS IPv6 Features and Supported Software Releases

The table below lists the IPv6 features supported in the 12.0S, 12.*x* T, 12.2S, 12.2SR, 12.2SX, 12.2SY, 12.3, 12.4, 15.0M, 15.0S, 15.0SY, 15.1S, 15.2S, 15.1T, and 15.2T Cisco IOS software release trains.

Note The table identifies the earliest release for each software release train in which the feature became available. Unless noted otherwise in the table, subsequent releases of that Cisco IOS software release train also support that feature.

Table 1 Supported IPv6 Feature

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*x*xSG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
IPv6							
IPv6: Base Protocols High Availability	Implementing IPv6 Addressing and Basic Connectivity	--	--	--	--	12.2(33)SRE	--
IPv6: CNS Agents for IPv6	Implementing IPv6 for Network Management	12.4(20)	15.0	--	12.2 (50)SG 3.2.0SG 15.0(2)SG	12.2(33)SRC	12.2 (50)SY

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*xx*SG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
IPv6 Device Tracking	Implementing First Hop Security in IPv6	--	--	--	--	--	12.2 (50)SY
Enhanced IPv6 Neighbor Discovery Cache Management	Implementing IPv6 Addressing and Basic Connectivity	--	--	--	--	--	12.2 (33)SXI7
IPv6: Full Selective Packet Discard Support	Implementing Selective Packet Discard in IPv6	15.1(3)	--	--	--	--	--
IPv6: HTTP(S) IPv6 Support (Infrastructure)	Implementing IPv6 for Network Management	12.4(20)	15.0	12.2 (44)SE	12.2 (44)SG 3.2.0SG 15.0(2)SG	12.2(33)SRC	12.2 (50)SY
IPv6: ICMP Rate Limiting	Implementing IPv6 Addressing and Basic Connectivity	12.2(8)	12.3	12.2 (25)SE	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (17a)SX1
IPv6: ICMPv6	Implementing IPv6 Addressing and Basic Connectivity	12.2(2)	12.3	12.2 (25)SEA	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	(17a)SX1
IPv6: ICMPv6 Redirect	Implementing IPv6 Addressing and Basic Connectivity	12.2(4)	12.3	12.2 (25)SEA	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (17a)SX1
IPv6: IPv6 ICMP RFC 4443	Implementing IPv6 Addressing and Basic Connectivity	12.4(9)T	--	12.2(52)	12.2 (50)SG 3.2.0SG 15.0(2)SG	--	12.2 (33)SXI

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*xx*SG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
IPv6: IP SLAs for IPv6	Implementing IPv6 for Network Management	12.4(20)	15.0	--	12.2 (50)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRC	12.2 (50)SY
IPv6 ACL Extensions for Mobile IPv6	Implementing Mobile IPv6	12.4(2)	--	--	--	12.2(33)SRB	12.2(33)SXI
IPv6: IPv6 Default Router Preferences	Implementing IPv6 Addressing and Basic Connectivity	12.4(2)	15.0	(46)	12.2 (46)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (33)SXH
IPv6: IPv6 for Config Logger	Implementing IPv6 for Network Management	12.4(20)	15.0	--	12.2 (50)SG 3.2.0SG 15.0(2)SG	12.2(33)SRC	12.2(50)SY
IPv6: IPv6 MTU Path Discovery	Implementing IPv6 Addressing and Basic Connectivity	12.2(2)	12.3	(25)SEA	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (17a)SX1
IPv6: IPv6 Neighbor Discovery	Implementing IPv6 Addressing and Basic Connectivity	12.2(2)	12.3	(25)SEA	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2(17a)SX1
IPv6: IPv6 NETCONF Support	Implementing IPv6 for Network Management	12.4(20)	15.0	--	12.2 (50)SG 3.2.0SG 15.0(2)SG	12.2(33)SRC	12.2 (50)SY
IPv6: IPv6 Stateless Autocon-figuration	Implementing IPv6 Addressing and Basic Connectivity	12.2(2)	12.3	(25)SEA	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (17a)SX1
IPv6: IPv6 Stateless Address Autoconfigur ation RFC 4862	Implementing IPv6 Addressing and Basic Connectivity	12.4(9)T	--	12.2(52)	12.2 (50)SG 3.2.0SG 15.0(2)SG	--	12.2 (33)SXI

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*xx*SG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
IPv6: IPv6 Static Cache Entry for Neighbor Discovery	Implementing IPv6 Addressing and Basic Connectivity	12.2(8)	12.3	(25)SEA	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (17a)SX1
IPv6: Per-Interface Neighbor Discovery Cache Limit	Implementing IPv6 Addressing and Basic Connectivity	15.1(3)	--	--	--	--	--
IPv6: IPv6 Support for TCL	Implementing IPv6 for Network Management	12.4(20)	15.0	--	12.2 (50)SG 3.2.0SG 15.0(2)SG	12.2(33)SRC	12.2(50)SY
IPv6: IPv6 Support in SOAP	Implementing IPv6 for Network Management	12.4(20)	15.0	--	12.2 (50)SG 3.2.0SG 15.0(2)SG	12.2(33)SRC	12.2 (50)SY
TACACS+ over IPv6	Implementing ADSL and Deploying Dial Access for IPv6	15.2(1)	15.1(1)S	(58)	--	15.1(1)S	12.2(33)SXJ
IPv6 VPN over MPLS	Implementing IPv6 VPN over MPLS	12.4(20)	15.0	--	--	12.2(33)SRB	12.2 (33)SXI
Mobile IPv6 Basic NEMO	Implementing Mobile IPv6	12.4(20)	15.0	--	--	--	--
Mobile IPv6 Home Agent	Implementing Mobile IPv6	12.3(14)	12.4	--	--	--	--
MPLS VPN 6VPE Support over IP Tunnels	Implementing IPv6 VPN over MPLS	--	--	--	--	12.2(33)SRB1	12.2 (33)SXI
BGP IPv6 PIC Edge for IP/MPLS	Implementing IPv6 VPN over MPLS	--	--	--	--	15.1(2)S	--

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*xx*SG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
BGP IPv6 Client for Single_Hop BFD	Configuring BGP Neighbor Session Options	--	--	--	--	15.1(2)S	--
IPv6: Neighbor Discovery Duplicate Address Detection	Implementing IPv6 Addressing and Basic Connectivity	12.2(4)	12.3	(25)SEA	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (17a)SX1
IPv6 ND Inspection	Implementing First Hop Security in IPv6	--	--	--	--	--	12.2 (50)SY
IPv6: Flexible NetFlow for IPv6 Replaces IPv6 NetFlow	Implementing NetFlow for IPv6	12.4(20)	15.0	--	--	--	--
IPv6: Ping	Implementing IPv6 for Network Management	12.2(2)	12.3	(25)SEA	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (17a)SX1
IPv6: syslog over IPv6	Implementing IPv6 for Network Management	12.4(4)	15.0	(44)	12.2 (44)SG 3.2.0SG 15.0(2)SG	12.2(33)SRB	12.2 (33)SXI
IPv6: Telnet, DNS, TFTP Client, Traceroute	Implementing IPv6 Addressing and Basic Connectivity, Implementing IPv6 for Network Management	12.2(2)	12.3	(25)SEA	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2(17a)SX1
IPv6: uRPF	Implementing IPv6 Addressing and Basic Connectivity	--	--	--	--	--	12.2 (50)SY

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*xx*SG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
IPv6 Address Types: Anycast	Implementing IPv6 Addressing and Basic Connectivity	12.3(4)	12.4	(25)SEA	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (33)SXH
IPv6 Address Types: Unicast	Implementing IPv6 Addressing and Basic Connectivity	12.2(2)	12.3	(25)SEA	--	12.2(33)SRA	12.2 (17a)SX1
IPv6 PACL Support	Implementing First Hop Security in IPv6	--	--	(46)	12.2 (54)SG 3.2.0SG 15.0(2)SG	--	12.2(33)SXI4
IPv6 RA Guard	Implementing First Hop Security in IPv6	--	--	--	12.2 (54)SG 3.2.0SG 15.0(2)SG	--	12.2 (33)SXI4
IPv6 Selective Packet Discard	Implementing Selective Packet Discard in IPv6	--	--	--	--	12.2(33)SRC	12.2 (33)SXH
IPv6 Support on BVI Interfaces	Implementing IPv6 Addressing and Basic Connectivity	15.1(2)	--	--	--	--	--
FTP IPv6 Support	Implementing IPv6 for Network Management	15.2(1)	--	--	--	15.1(3)S	--
IPv6 Switching Services							
CEFv6 Switched Configured IPv6 over IPv4 Tunnels	Implementing Tunneling for IPv6	15.1(3)	--	--	--	--	--

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*xx*SG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
CEFv6 Switched Configured IPv6 over IPv6 GRE Tunnels	Implementing Tunneling for IPv6	15.1(3)	--	--	--	--	--
IPv6 Switching: CEFv6 Switched Automatic IPv4-Compatible Tunnels	Implementing Tunneling for IPv6	12.3(2)	12.4	--	--	12.2(33)SRA	12.2(17a)SX1
IPv6 Switching: CEFv6 Switched Configured IPv6 over IPv4 Tunnels	Implementing Tunneling for IPv6	12.2(13)	12.4	--	--	12.2(33)SRA	12.2(18)SXE
IPv6 Switching: CEFv6 Switched ISATAP	Implementing Tunneling for IPv6	12.3(2)	12.4	--	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2(17a)SX1
IPv6 Switching: Cisco Express Forwarding/ Distributed Cisco Express Forwarding Support	Implementing IPv6 Addressing and Basic Connectivity	12.2(13)	12.3	(25)SEA	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (17a)SX1
IPv6 Switching: Provider Edge Router over MPLS (6PE)	Implementing IPv6 over MPLS	12.2(15)	12.3	--	--	12.2(33)SRA	12.2(17b)SXA
IPv6 Routing							

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*xx*SG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
BFD IPv6 Encapsulation Support	Implementing Bidirectional Forwarding Detection for IPv6	15.1(2)	--	--	--	12.2(33)SRE	15.0(1)SY
BFD Support for EIGRP IPv6	IP Routing: BFD Configuration Guide	--	--	--	--	15.2(2)S	--
EIGRP IPv6 VRF Lite	Implementing EIGRP for IPv6	--	--	--	--	15.1(1)S	15.0(1)SY1
IPv6 Routing: EIGRP Support	Implementing EIGRP for IPv6	12.4(6)	--	(40)	12.2 (40)SG 3.2.0SG 15.0(2)SG	12.2(33)SRB	12.2 (33)SXI
IPv6 Policy-Based Routing	Implementing Policy-Based Routing for IPv6	12.3(7)	12.4	--	--	15.2(1)S	12.2 (33)SXI4
IPv6 Routing: IS-IS Multitopology Support for IPv6	Implementing IS-IS for IPv6	12.2(15)	12.3	--	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (18)SXE
IPv6 Routing: IS-IS Support for IPv6	Implementing IS-IS for IPv6	12.2(8)	12.3	--	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	(17a)SX1
IPv6 IS-IS Local RIB	Implementing IS-IS for IPv6	12.3(4)T	12.4	--	--	12.2(33)SRA	12.2 (33)SXH
IPv6 Routing: Multiprotocol BGP Extensions for IPv6	Implementing Multiprotocol BGP for IPv6	12.2(2)	12.3	--	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (17a)SX1

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*xx*SG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
IPv6 Routing: Multiprotocol BGP Link-Local Address Peering	Implementing Multiprotocol BGP for IPv6	12.2(4)	12.3	--	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2(17a)SX1
IPv6: NSF and Graceful Restart for MP-BGP IPv6 Address Family	Implementing Multiprotocol BGP for IPv6	--	--	--	--	12.2(33)SRE	15.0(1)SY
OSPFv3 Fast Convergence: LSA and SPF Throttling	Implementing OSPFv3	--	15.0(1)M	--	--	12.2(33)SRC	15.0(1)SY
IPv6 Routing: OSPFv3	Implementing OSPFv3	12.2(15)	12.3	(25)SEA	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (17a)SX1
IPv6 Routing: OSPF for IPv6 Authentication Support with IPsec	Implementing OSPFv3	12.3(4)	12.4	--	--	15.2(1)S	15.0(1)SY1
OSPFv3 IPSec ESP Encryption and Authentication	Implementing OSPFv3	12.4(9)	15.0	--	--	--	15.0(1)SY1
IPv6 Routing: RIP for IPv6 (RIPng)	Implementing RIP for IPv6	12.2(2)	12.3	(25)SEA	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (17a)SX1
IPv6: RIPng Nonstop Forwarding	Implementing RIP for IPv6	--	--	--	--	12.2(33)SRE	15.0(1)SY

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*xx*SG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
IPv6 Routing: Route Redistribution	Implementing IS-IS for IPv6, Implementing RIP for IPv6	12.2(2)	12.3	(25)SEA	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (18)SXE
IPv6 Routing: Static Routing	Implementing Static Routes for IPv6	12.2(2)	12.3	(25)SEA	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (17a)SX1
OSPFv3 Address Families	Implementing OSPFv3	15.2(1)	--	--	--	15.1(3)S	--
OSPFv3 Dynamic Interface Cost Support	Implementing OSPFv3	12.4(15)	15.0	--	--	--	--
OSPFv3 External Path Preference Option	Implementing OSPFv3	15.2(3)	--	--	--	15.1(3)S	--
OSPFv3 for BFD	Implementing OSPFv3, Implementing Bidirectional Forwarding Detection for IPv6	15.1(2)	--	--	--	12.2(33)SRE	15.0(1)SY
OSPFv3 Graceful Restart	Implementing OSPFv3	--	15.0(1)M	(58)	--	12.2(33)SRE	15.0(1)SY
OSPFv3 Manet Extensions	IP Mobility Guide	15.2(1)	--	--	--	--	--
OSPFv3 Max-Metric Router LSA	Implementing OSPFv3	15.2(3)	--	--	--	15.1(3)S	--
Static Route Support for BFD over IPv6	Implementing Bidirectional Forwarding Detection for IPv6	15.1(2)	--	--	--	--	15.0(1)SY1

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*xx*SG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
VRF Lite Support for IPv6	Implementing Multiprotocol BGP for IPv6	--	--	(58)	--	--	--
IPv6 Services and Management							
Hardware and Software Counters Granularity for IPv4 and IPv6 ACL Statistics	Implementing Traffic Filters and Firewalls for IPv6 Security	--	--	--	--	--	12.2 (50)SY
IPsec IPv6 Phase 2 Support	Implementing IPSec in IPv6 Security	12.4(4)	15.0	--	--	--	--
IPv6 Secure Neighbor Discovery (SeND)	Implementing First Hop Security in IPv6	12.4(24)	15.0	--	--	--	--
IPv6 Services: AAAA DNS Lookups over an IPv4 Transport	Implementing IPv6 Addressing and Basic Connectivity	12.2(2)	12.3	(25)SEA	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (17a)SX1
IPv6 Services: Cisco Discovery Protocol: IPv6 Address Family Support for Neighbor Information	Implementing IPv6 Addressing and Basic Connectivity	12.2(8)	12.3	(25)SEE	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (18)SXE
IPv6 Services: CISCO-IP-FORWARDING-MIB Support	Implementing IPv6 for Network Management	12.2(15)	12.3	(25)SEE	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (18)SXE

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*xx*SG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
IPv6 Services: CISCO-IP-MIB Support	Implementing IPv6 for Network Management	12.2(15)	12.3	(25)SEE	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (18)SXE
IPv6 Services: DNS Lookups over an IPv6 Transport	Implementing IPv6 Addressing and Basic Connectivity	12.2(8)	12.3	(25)SED	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRE2	12.2 (17a)SX1
IPv6 Services: Extended Access Control Lists	Implementing Traffic Filters and Firewalls for IPv6 Security	12.2(13)	12.3	(25)SED	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (17a)SX1
FHRP: GLBP for IPv6	Configuring First Hop Redundancy Protocols in IPv6	12.4(6)	15.0	(58)	--	--	12.2 (33)SXI
IPv6 Services: Generic Prefix	Implementing IPv6 Addressing and Basic Connectivity	12.3(4)	12.4	--	--	--	--
HSRP for IPv6	Configuring First Hop Redundancy Protocols in IPv6	12.4(4)	15.0	(46)	12.2 (52)SG 3.2.0SG 15.0(2)SG	12.2(33)SRB	12.2 (33)SXI
HSRP: Global IPv6 Address	Configuring First Hop Redundancy Protocols in IPv6	--	--	--	--	--	12.2 (33)SXI4
SSO: HSRP	Configuring First Hop Redundancy Protocols in IPv6	--	--	--	12.2 (52)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (33)SXH

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*x*xSG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
ISSU: HSRP	Configuring First Hop Redundancy Protocols in IPv6	--	--	--	12.2 (52)SG 3.2.0SG 15.0(2)SG	--	12.2 (33)SXI
IOS Zone-Based Firewall	Implementing Traffic Filters and Firewalls for IPv6 Security	15.1(2)	--	--	--	--	--
IPv6 ACL Extensions for IPsec Authentication Header	Implementing Traffic Filters and Firewalls for IPv6 Security	12.4(20)	15.0	--	--	--	--
IPv6 ACL Extensions for Hop by Hop Filtering	IPv6 ACL Extensions for Hop by Hop Filtering	15.2(3)	--	--	--	15.2(2)S	--
IPv6 IOS Firewall	Implementing Traffic Filters and Firewalls for IPv6 Security	12.3(7)	12.4	--	--	--	--
IPv6 IOS Firewall FTP Application Support	Implementing Traffic Filters and Firewalls for IPv6 Security	12.3(11)	--	--	--	--	--
IPv6 IPsec VPN	Implementing IPSec in IPv6 Security	12.4(4)	15.0	--	--	--	--
IPv6 Remote Access for IPSec VPN	FlexVPN and Internet Key Exchange Version 2 Configuration Guide	15.2(3)	--	--	--	--	--

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*xx*SG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
IPv6 over DMVPN	Implementing Dynamic Multipoint VPN over IPv6	12.4(20)	15.0	--	--	--	--
IPv6 Transport for DMVPN	Implementing Dynamic Multipoint VPN over IPv6	15.2(1)	--	--	--	--	--
IPv6 Services: RFC 4293 IP-MIB (IPv6 Only) and RFC 4292 IP-FORWARD-MIB (IPv6 Only)	Implementing IPv6 for Network Management	15.1(3)	--	(58)	12.2 (54)SG 3.2.0SG 15.0(2)SG	12.2(33)SRC	12.2 (50)SY
IPv6 Services: Secure Shell (SSH) Support over IPv6	Implementing IPv6 for Network Management	12.2(8)	12.3	(25)SEE	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (17a)SX1
IPv6 Services: SNMP over IPv6	Implementing IPv6 for Network Management	12.3(14)	12.4	(44)	12.2 (44)SG 3.2.0SG 15.0(2)SG	12.2(33)SRB	12.2(33)SXI
IPv6 Services: Standard Access Control Lists	Implementing Traffic Filters and Firewalls for IPv6 Security	12.2(2)	12.3	(25)SED	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2(17a)SX1
IKEv2 Headend Support for Remote Access Clients	http://www.cisco.com/en/US/docs/ios-xml/ios/sec_conn_ikevpn/configuration/15-2mt/sec-key-exch-ver2.html	15.2(1)	--	--	--	--	--

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*x*xSG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
NBAR IPv6 Transition Mechanism Detection		15.1(3)	--	--	--	--	--
NTPv4	Implementing NTPv4 in IPv6	12.4(20)	--	12.2(58)SE	--	15.1(2)S	12.2 (33)SXJ
SNMPv3 - 3DES and AES Encryption Support	Implementing IPv6 for Network Management	12.4(2)	15.0	(52)	12.2 (50)SG 3.2.0SG 15.0(2)SG	12.2(33)SRB	12.2(33)SXI
IPv6 Broadband Access							
Broadband IPv6 Counter Support at LNS	Implementing ADSL and Deploying Dial Access for IPv6	12.2(13)	12.3	--	--	12.2(33)SRC	--
IPv6 Access Services: AAA Support for Cisco VSA IPv6 Attributes	Implementing ADSL and Deploying Dial Access for IPv6	12.2(13)	12.3	--	--	12.2(33)SRC	--
IPv6 Access Services: AAA Support for RFC 3162 IPv6 RADIUS Attributes	Implementing ADSL and Deploying Dial Access for IPv6	12.3(4)	12.4	(58)	--	12.2(33)SRC	--
IPv6 Access Services: PPPoA	Implementing ADSL and Deploying Dial Access for IPv6	12.2(13)	12.3	--	--	12.2 (33)SRC	--
IPv6 Access Services: PPPoE	Implementing ADSL and Deploying Dial Access for IPv6	12.2(13)	12.3	--	--	12.2(33)SRC	--

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*xx*SG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
IPv6 Access Services: Prefix Pools	Implementing ADSL and Deploying Dial Access for IPv6	12.2(13)	12.3	--	--	12.2 (33)SRC	--
IPv6 Access Services: RBE	Implementing IPv6 Addressing and Basic Connectivity	12.3(4)	12.4	--	--	12.2 (33)SRC	--
RADIUS over IPv6	Implementing ADSL and Deploying Dial Access for IPv6	15.2(1)	--	(58)	--	--	--
DHCP for IPv6							
DHCP: DHCPv6 Individual Address Assignment	Implementing DHCP for IPv6	12.4(24)	--	(46)	--	--	--
DHCPv6 Relay SSO/ ISSU	Implementing DHCP for IPv6	--	--	--	--	12.2 (33)SRE	--
DHCPv6 Bulk Lease Query	Implementing DHCP for IPv6	--	--	(58)	--	15.1(1)S	--
DHCPv6 Relay: Source Configuration	Implementing DHCP for IPv6	--	--	(58)	--	12.2 (33)SRE	--
IPv6 Access Services: DHCP for IPv6 Relay Agent	Implementing DHCP for IPv6	12.3(11)	12.4	(46)	12.2 (50)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRC	12.2(33)SXI

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*xx*SG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
IPv6 Access Services: DHCPv6 Client Information Refresh Option	Implementing DHCP for IPv6	12.4(15)	15.0	--	--	--	--
IPv6 Access Services: DHCPv6 Ethernet Remote ID Option	Implementing DHCP for IPv6	--	--	(46)	12.2 (52)SG 3.2.0SG 15.0(2)SG	12.2(33)SRC	12.2 (33)SXI
IPv6 Access Services: DHCPv6 Prefix Delegation	Implementing DHCP for IPv6, Implementing ADSL and Deploying Dial Access for IPv6	12.3(4)	12.4	--	--	12.2 (33)SRA	12.2 (18)SXE
IPv6 Access Services: DHCPv6 Prefix Delegation via AAA	Implementing ADSL and Deploying Dial Access for IPv6	12.3(14)	12.4	--	--	--	--
IPv6 Access Services: DHCPv6 Relay Agent Notification for Prefix Delegation	Implementing DHCP for IPv6	--	--	(46)	12.2 (52)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRC	12.2(33)SXI
IPv6 Access Services: DHCPv6 Relay - Reload Persistent Interface ID Option	Implementing DHCP for IPv6	--	--	(46)	12.2 (52)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRC	12.2 (33)SXI

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*xx*SG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
IPv6 Access Services: DHCPv6 Server Stateless Auto Configuration	Implementing DHCP for IPv6	12.4(15)	--	(46)	12.2 (52)SG 3.2.0SG 15.0(2)SG	(33)SRC	--
IPv6 Access Services: Stateless DHCPv6	Implementing DHCP for IPv6	12.3(4)	12.4	--	--	12.2(33)SRA	12.2 (18)SXE
DHCPv6 Server: MPLS VPN Support	Implementing DHCP for IPv6	--	--	--	--	15.1(2)S	--
DHCPv6 Relay: MPLS VPN Support	Implementing DHCP for IPv6	--	--	--	--	15.1(2)S	--
DHCPv6 Server: Relay-Client Support in a VRF Lite Environment	Implementing DHCP for IPv6	--	--	(58)	--	--	--
IPv6 Multicast							
IPv6 Multicast: Address Family Support for Multiprotocol Border Gateway Protocol (MBGP)	Implementing IPv6 Multicast	12.3(4)	12.4	--	--	12.2 (33)SRA	12.2 (33)SXH
IPv6 Multicast: Address Group Range Support	Implementing IPv6 Multicast	--	15.0(1)M	--	12.2 (40)SG 3.2.0SG 15.0(2)SG	12.2(33)SRE	12.2 (33)SXI

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*xx*SG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
IPv6 Multicast: Bandwidth-Based Call Admission Control (CAC)	Implementing IPv6 Multicast	--	--	--	12.2 (40)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRE	--
IPv6 Multicast: Explicit Tracking of Receivers	Implementing IPv6 Multicast	12.3(7)	12.4	--	12.2 (40)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2 (33)SXH
IPv6 Multicast: IPv6 Bidirectional PIM	Implementing IPv6 Multicast	12.3(7)	12.4	--	12.2 (40)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	--
IPv6 Multicast: IPv6 BSR	Implementing IPv6 Multicast	12.3(11)	12.4	--	12.2 (40)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (18)SXE
IPv6 Multicast: IPv6 BSR: Ability to Configure RP Mapping	Implementing IPv6 Multicast	12.4(2)	--	--	12.2 (40)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRE	12.2 (50)SY
IPv6 Multicast: IPv6 BSR Bidirectional Support	Implementing IPv6 Multicast	12.3(14)	12.4	--	12.2 (40)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRE	--
IPv6 Multicast: IPv6 BSR Scoped-Zone Support	Implementing IPv6 Multicast	12.3(14)	12.4	--	12.2 (40)SG 3.2.0SG 15.0(2)SG	--	--
IPv6 Multicast: MFIB Display Enhancements	Implementing IPv6 Multicast	12.3(7)	12.4	--	12.2 (40)SG 3.2.0SG 15.0(2)SG	--	--

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*xx*SG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
IPv6 Multicast: Multicast Listener Discovery (MLD) Protocol, versions 1 and 2	Implementing IPv6 Multicast	12.3(2)	12.4	--	12.2 (40)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2 (18)SXE
IPv6 Multicast: MLD Access Group	Implementing IPv6 Multicast	12.3(4)	12.4	--	12.2 (40)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2 (33)SXH
IPv6 Multicast: MLD Group Limits	Implementing IPv6 Multicast	12.4(2)	--	--	12.2 (40)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRE	12.2 (50)SY
IPv6 Multicast: MLD Proxy	Implementing IPv6 Multicast	15.1(2)	--	--	--	--	--
IPv6 Multicast: MLD Snooping	Implementing IPv6 Multicast	--	--	(25)SED	12.2 (40)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRB	12.2 (18)SXE
IPv6 Multicast: Multicast User Authentication and Profile Support	Implementing IPv6 Multicast	12.4(4)	--	--	12.2 (40)SG 3.2.0SG 15.0(2)SG	--	--
IPv6 Multicast: PIM Accept Register	Implementing IPv6 Multicast	12.3(4)	12.4	--	12.2 (40)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2 (33)SXH
IPv6 Multicast: PIM Source Specific Multicast (PIM-SSM)	Implementing IPv6 Multicast	12.3(2)	12.4	--	12.2 (40)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2 (18)SXE

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*xx*SG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
IPv6 Multicast: PIM Sparse Mode (PIM-SM)	Implementing IPv6 Multicast	12.3(2)	12.4	--	12.2 (40)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (18)SXE
IPv6 Multicast: Scope Boundaries	Implementing IPv6 Multicast	12.3(2)	12.4	--	12.2 (40)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2 (18)SXE
IPv6 Multicast: PIM Embedded RP Support	Implementing IPv6 Multicast	12.3(4)	12.4	--	12.2 (40)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2 (33)SXH
IPv6 Multicast: Routable Address Hello Option	Implementing IPv6 Multicast	12.3(4)	12.4	--	12.2 (40)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2(33)SXH
IPv6 Multicast: RPF Flooding of Bootstrap Router (BSR) Packets	Implementing IPv6 Multicast	12.3(4)	12.4	--	12.2 (40)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2 (33)SXH
IPv6 Multicast: SSM Mapping for MLDv1 SSM	Implementing IPv6 Multicast	12.4(2)	--	--	12.2 (40)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2 (18)SXE
IPv6 Multicast: Static Multicast Routing (mroute)	Implementing IPv6 Multicast	12.3(4)	12.4	--	12.2 (40)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2 (33)SXH
IPv6 Multicast VRF Lite	Implementing IPv6 Multicast	--	15.1(4)M	--	--	--	--
ISSU: IPv6 Multicast	Implementing IPv6 Multicast	--	--	--	--	--	15.0(1)SY

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*xx*SG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
NSF/SSO: IPv6 Multicast	Implementing IPv6 Multicast	--	--	--	--	12.2 (33)SRE	15.0(1)SY
PIMv6: Anycast RP Solution	PIMv6-- Anycast RP Solution	15.2(3)	--	--	--	15.1(3)S	--
NAT Protocol Translation (NAT-PT)		12.2 (13)	12.3	--	--	--	
NAT-PT: Support for DNS ALG	Implementing NAT Protocol Translation	12.2(13)	12.3	--	--	--	--
NAT-PT: Support for FTP ALG	Implementing NAT Protocol Translation	12.3(2)	12.4	--	--	--	--
NAT-PT: Support for Fragmentation	Implementing NAT Protocol Translation	12.3(2)	12.4	--	--	--	--
NAT-PT: Support for Overload (PAT)	Implementing NAT Protocol Translation	12.3(2)	12.4	--	--	--	--
IPv6 Tunnel Services							
IPv6 Rapid Deployment	Implementing Tunneling for IPv6	15.1(3)	--	--	--	--	--
IPv6 Tunneling: Automatic 6to4 Tunnels	Implementing Tunneling for IPv6	12.2(2)	12.3	--	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2 (18)SXE
IPv6 Tunneling: Automatic IPv4-Compatible Tunnels	Implementing Tunneling for IPv6	12.2(2)	12.3	--	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2 (18)SXE

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*x*xSG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
IPv6 Tunneling: CEF Switched Automatic 6to4 Tunnels	Implementing Tunneling for IPv6	12.3(2)	12.3	--	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2 (18)SXE
IPv6 Tunneling: CLNS Support for GRE IPv6 and IPv4 Tunnels	Implementing Tunneling for IPv6	12.3(7)	12.4	--	--	12.2(33)SRA	12.2(33)SXH
IPv6 Tunneling: IP over IPv6 GRE Tunnels	Implementing Tunneling for IPv6	12.3(7)	12.4	--	--	--	--
IPv6 Tunneling: IPv6 over IPv4 GRE Tunnels	Implementing Tunneling for IPv6	12.2(4)	12.3	--	--	12.2 (33)SRA	12.2 (17a)SX1
IPv6 Tunneling: IPv6 over IPv6 Tunnels	Implementing Tunneling for IPv6	12.3(7)	12.4	--	--	--	--
IPv6 Tunneling: ISATAP Tunnel Support	Implementing Tunneling for IPv6	12.2(15)	12.3	--	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2 (17a)SX1
IPv6 Tunneling: Manually Configured IPv6 over IPv4 Tunnels	Implementing Tunneling for IPv6	12.2(2)	12.3	--	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2 (17a)SX1
mGRE Tunnel support over IPv6	Implementing Tunneling for IPv6	15.2(1)	--	--	--	--	--

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*xx*SG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
IPv6 Quality of Service (QoS)							
IPv6: QoS Trust	Configuring QoS	--	--	(52)	12.2 (50)SG 3.2.0SG 15.0(2)SG	--	--
IPv6 QoS: MQC Packet Classification	Implementing QoS for IPv6	12.2(13)	12.3	--	12.2 (50)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2 (18)SXE
IPv6 QoS: MQC Packet Marking/Re-Marking	Implementing QoS for IPv6	12.2(13)	12.3	--	12.2 (50)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2 (18)SXE
IPv6 QoS: MQC Traffic Policing	Implementing QoS for IPv6	12.2(13)	12.3	--	12.2 (50)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2 (18)SXE
IPv6 QoS: MQC Traffic Shaping	Implementing QoS for IPv6	12.2(13)	12.3	--	12.2 (50)SG 3.2.0SG 15.0(2)SG	12.2(33)SRA	12.2 (18)SXE
IPv6 QoS: MQC Weighted Random Early Detection (WRED)-Based Drop	Implementing QoS for IPv6	12.2(13)	12.3	--	12.2 (50)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2 (18)SXE
IPv6 QoS: Queueing	Implementing QoS for IPv6	12.2(13)	12.3	--	12.2 (50)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2 (18)SXE
IPv6 Voice							
Cisco UBE RTCP Voice Pass-Through for IPv6		15.2(1)	--	--	--	--	--

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*xx*SG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
VoIP Over IPv6	Implementing Voice over IPv6	12.4(22)	--	--	--	--	--
T.38 Fax Support on Cisco UBE for IPv6		15.2(1)	--	--	--	--	--
IPv6 Data Link Layer							
IPv6 Data Link: ATM PVC and ATM LANE	Implementing IPv6 Addressing and Basic Connectivity	12.2(2)	12.3	--	--	12.2 (33)SRA	--
IPv6 Data Link: Ethernet, Fast Ethernet, Gigabit Ethernet, and 10-Gigabit Ethernet	Implementing IPv6 Addressing and Basic Connectivity	12.2(2)	12.3	(25)SEA	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	--
IPv6 Data Link: Frame Relay PVC	Implementing IPv6 Addressing and Basic Connectivity	12.2(2)	12.3	--	--	12.2 (33)SRA	--
IPv6 Data Link: High-Level Data Link Control	Implementing IPv6 Addressing and Basic Connectivity	12.2(2)	12.3	--	--	12.2 (33)SRA	--

Feature	Location	12.*x* T/ 15.*x* T Release	12.*x* /15.*x* Release	12.2SE Release	12.2SG, 15.*xx*SG 3.*x* SG Release	12.2SR/ 15S Release	12.2SX, 12.2SY, 15.0SY Release
IPv6 Data Link: PPP Service over Packet over SoNET, ISDN, and Serial (Synchronous and Asynchronous) Interfaces	Implementing IPv6 Addressing and Basic Connectivity	12.2(2)	12.3	--	--	12.2 (33)SRA	--
IPv6 Data Link: VLANs Using Cisco Inter-Switch Link (ISL)	Implementing IPv6 Addressing and Basic Connectivity	12.2(2)	12.3	(25)SEA	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2(18)SXE
IPv6 Data Link: VLANs Using IEEE 802.1Q Encapsulation	Implementing IPv6 Addressing and Basic Connectivity	12.2(2)	12.3	(25)SEA	12.2 (25)SG 3.2.0SG 15.0(2)SG	12.2 (33)SRA	12.2 (18)SXE

Cisco Platforms Supporting IPv6 Hardware Forwarding

Supported Platforms

The table below lists the Cisco platforms that have IPv6 hardware forwarding and the Cisco IOS software release trains that introduce the feature.

Note The table lists only the Cisco IOS software release that introduced support for a given feature in a given Cisco IOS software release train. Unless noted otherwise in the table, subsequent releases of that Cisco IOS software release train also support that feature.

Table 2 Minimum Required Release for Cisco Platforms Supporting IPv6 Hardware Forwarding

Hardware and Feature	Cisco IOS Software Release
Cisco 12000 Series	
IP ISE line card IPv6 forwarding	12.0(23)S
IP ISE line card extended ACLs	12.0(25)S
IP ISE line card IPv6 over MPLS (6PE)	12.0(25)S
IP ISE line card IPv6 Multicast assist	12.0(26)S
IP ISE line card IPv6 QoS	12.0(28)S
Engine 5 line card IPv6 hardware forwarding	12.0(31)S
IP Receive ACL for IPv6 traffic	12.0(32)S
Cisco 10000 Series	
Cisco 10000 series Performance Routing Engine 2 (PRE-2)	12.2(28)SB
Cisco 10000 series PRE-3	12.2(31)SB
Cisco 10000 series 6PE support	12.2(31)SB
Cisco 10000 series PRE-4	12.2(33)SB
Cisco 10720 Series	
PxF accelerated for IPv6 forwarding	12.0(26)S, 12.2(28)SB
PxF accelerated for IPv6 extended ACLs	12.0(26)S
PxF accelerated for IPv6 over MPLS (6PE)	12.0(26)S
PRE-2 hardware forwarding	12.2(28)SB
Cisco 7600 Series, Cisco Catalyst 6500, Cisco Catalyst 3700, and Cisco Catalyst 3500	
IPv6: Express setup	12.2(35)SE
Cisco Catalyst 3560 series	12.2(25)SEA
Cisco Catalyst 3750 series	12.2(25)SEA
IPv6: IPv6 and IPv4 TCAM templates	12.2(25)SEA
IPv6: IPv6 neighbor discovery throttling	12.2(25)SEA
Cisco Catalyst 3560E series	12.2(35)SE2
Cisco Catalyst 3570E series	12.2(35)SE2

Hardware and Feature	Cisco IOS Software Release
Cisco Catalyst 3560 series: IPv6 multicast hardware layer	12.2(25)SED
Supervisor Engines 720 and 720-3bxl	12.2(33)SRA
Route/switch processor 720 on Cisco 7600 series	12.2(33)SRB
Supervisor Engine 720 IPv6 forwarding	12.2(17a)SX1
Supervisor Engine 720 IPv6 extended ACLs	12.2(17a)SX1
Supervisor Engine 720 IPv6 over MPLS (6PE)	12.2(17b)SXA
Supervisor Engine 720 IPv6 multicast hardware forwarding	12.2(18)SXE
Supervisor Engine 720 IPv6 multicast RPR/RPR+ support	12.2(18)SXE
Supervisor Engine 720 IPv6 multicast hardware-assisted egress replication	12.2(18)SXE
Supervisor Engine 32/MSFC2A	12.2(18)SXF

Additional 12.2S Release Trains

Several early-deployment Cisco IOS software Release 12.2S trains synchronize to the Cisco IOS software mainline Release 12.2S train. The following table lists information about the release trains on which IPv6 hardware is used.

Table 3 Minimum Required Release for IPv6 Hardware on Early-Deployment 12.2S Cisco IOS Software Release Trains

Early-Deployment Cisco IOS Software Release and Hardware	Release Description
12.2(28)SB and 12.2(33)SB on Cisco 10000 series	Not all features for Cisco IOS Release 12.2(28)SB or Cisco IOS Release 12.2(33)SB are supported on the Cisco 10000 series routers. For further information on Cisco IOS Release 12.2(28)SB or Cisco IOS Release 12.2(33)SB, see the release notes at the following URLs: http://www.cisco.com/en/US/products/ps6566/prod_release_notes_list.html
12.2(25)SEA on Cisco Catalyst 3560 and 3570 series	12.2(25)SEA supports a subset of the 12.2S IPv6 feature set. IPv6 multicast is not supported.
12.2(33)SRA on Cisco 7600 series	12.2(33)SRA includes all IPv6 features from Cisco IOS software releases 12.2S and 12.2SX.
12.2SX on Cisco Catalyst 6500	12.2(17)SX includes the entire Cisco IOS software Release 12.2(14)S feature set, plus OSPFv3.
12.2(17d)SXB on Cisco Catalyst 6500 Supervisor Engine 2/MSFC2	IPv6 support provided on 12.2(17)SXB for Cisco Catalyst 6500 Supervisor Engine 2/MSFC2.

Early-Deployment Cisco IOS Software Release and Hardware	Release Description
12.2(18)SXE on Cisco Catalyst 6500 and Cisco 7600 series	12.2(18)SXE supports IPv6 multicast hardware forwarding.
12.2(18)SXF on Supervisor Engine 32/MSFC2A	NA
12.2(35)SE2 on Cisco Catalyst 3560E and 3570E series	NA
12.2(40)SE on Cisco Catalyst 2960	IPv6 support provided for MLD snooping.
12.2(33)SCA on UBR	Support is provided for DHCPv6 relay agent notification for prefix delegation.

Additional References

Related Documents

Related Topic	Document Title
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*

RFCs

RFCs	Title
RFC 1195	*Use of OSI IS-IS for Routing in TCP/IP and Dual Environments*
RFC 1267	*A Border Gateway Protocol 3 (BGP-3)*
RFC 1305	*Network Time Protocol (Version 3) Specification, Implementation and Analysis*
RFC 1583	*OSPF version 2*
RFC 1772	*Application of the Border Gateway Protocol in the Internet*
RFC 1886	DNS Extensions to Support IP version 6
RFC 1918	*Address Allocation for Private Internets*
RFC 1981	*Path MTU Discovery for IP version 6*
RFC 2080	*RIPng for IPv6*
RFC 2281	*Cisco Hot Standby Router Protocol (HSRP)*
RFC 2332	*NBMA Next Hop Resolution Protocol (NHRP)*

RFCs	Title
RFC 2373	*IP Version 6 Addressing Architecture*
RFC 2374	*An Aggregatable Global Unicast Address Format*
RFC 2375	*IPv6 Multicast Address Assignments*
RFC 2401	*Security Architecture for the Internet Protocol*
RFC 2402	*IP Authentication Header*
RFC 2404	The Use of Hash Message Authentication Code Federal Information Processing Standard 180-1 within Encapsulating Security Payload and Authentication Header
RFC 2406	IP Encapsulating Security Payload (ESP)
RFC 2407	The Internet Security Domain of Interpretation for ISAKMP
RFC 2408	Internet Security Association and Key Management Protocol
RFC 2409	Internet Key Exchange (IKE)
RFC 2427	*Multiprotocol Interconnect over Frame Relay*
RFC 2428	*FTP Extensions for IPv6 and NATs*
RFC 2460	*Internet Protocol, Version 6 (IPv6) Specification*
RFC 2461	*Neighbor Discovery for IP Version 6 (IPv6)*
RFC 2462	*IPv6 Stateless Address Autoconfiguration*
RFC 2463	*Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification*
RFC 2464	*Transmission of IPv6 Packets over Ethernet*
RFC 2467	*Transmission of IPv6 Packets over FDDI*
RFC 2472	*IP Version 6 over PPP*
RFC 2473	*Generic Packet Tunneling in IPv6 Specification*
RFC 2474	*Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*
RFC 2475	*An* Architecture for Differentiated Services Framework
RFC 2492	IPv6 over ATM

RFCs	Title
RFC 2545	*Use of BGP-4 Multiprotocol Extensions for IPv6 Inter-Domain Routing*
RFC 2590	*Transmission of IPv6 Packets over Frame Relay Specification*
RFC 2597	Assured Forwarding PHB
RFC 2598	*A* n Expedited Forwarding PHB
RFC 2640	*Internet Protocol, Version 6 Specification*
RFC 2684	*Multiprotocol Encapsulation over ATM Adaptation Layer 5*
RFC 2697	A Single Rate Three Color Marker
RFC 2698	A Two Rate Three Color Marker
RFC 2710	*Multicast Listener Discovery (MLD) for IPv6*
RFC 2711	IPv6 Router Alert Option
RFC 2732	*Format for Literal IPv6 Addresses in URLs*
RFC 2765	*Stateless IP/ICMP Translation Algorithm (SIIT)*
RFC 2766	*Network Address Translation-Protocol Translation (NAT-PT)*
RFC 2858	*Multiprotocol Extensions for BGP-4*
RFC 2893	*Transition Mechanisms for IPv6 Hosts and Routers*
RFC 3056	Connection of IPv6 Domains via IPv4 Clouds
RFC 3068	*An Anycast Prefix for 6to4 Relay Routers*
RFC 3095	*RObust Header Compression (ROHC): Framework and Four Profiles: RTP, UDP, ESP, and Uncompressed*
RFC 3107	*Carrying Label Information in BGP-4*
RFC 3137	OSPF Stub Router Advertisement
RFC 3147	*Generic Routing Encapsulation over CLNS*
RFC 3152	*Delegation of IP6.ARPA*
RFC 3162	*RADIUS and IPv6*
RFC 3315	*Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*

RFCs	Title
RFC 3319	*Dynamic Host Configuration Protocol (DHCPv6) Options for Session Initiated Protocol (SIP) Servers*
RFC 3392	*Capabilities Advertisement with BGP-4*
RFC 3414	*User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)*
RFC 3484	*Default Address Selection for Internet Protocol version 6 (IPv6)*
RFC 3513	*Internet Protocol Version 6 (IPv6) Addressing Architecture*
RFC 3576	Change of Authorization
RFC 3587	*IPv6 Global Unicast Address Format*
RFC 3590	*Source Address Selection for the Multicast Listener Discovery (MLD) Protocol*
RFC 3596	*DNS Extensions to Support IP Version 6*
RFC 3633	*DHCP IPv6 Prefix Delegation*
RFC 3646	*DNS Configuration options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*
RFC 3697	*IPv6 Flow Label Specification*
RFC 3736	*Stateless DHCP Service for IPv6*
RFC 3756	*IPv6 Neighbor Discovery (ND) Trust Models and Threats*
RFC 3759	*RObust Header Compression (ROHC): Terminology and Channel Mapping Examples*
RFC 3775	*Mobility Support in IPv6*
RFC 3810	*Multicast Listener Discovery Version 2 (MLDv2) for IPv6*
RFC 3846	*Mobile IPv4 Extension for Carrying Network Access Identifiers*
RFC 3879	*Deprecating Site Local Addresses*
RFC 3898	*Network Information Service (NIS) Configuration Options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*

RFCs	Title
RFC 3954	*Cisco Systems NetFlow Services Export Version 9*
RFC 3956	*Embedding the Rendezvous Point (RP) Address in an IPv6 Multicast Address*
RFC 3963	*Network Mobility (NEMO) Basic Support Protocol*
RFC 3971	SEcure Neighbor Discovery (SEND)
RFC 3972	Cryptographically Generated Addresses (CGA)
RFC 4007	IPv6 Scoped Address Architecture
RFC 4075	*Simple Network Time Protocol (SNTP) Configuration Option for DHCPv6*
RFC 4087	IP Tunnel MIB
RFC 4091	*The Alternative Network Address Types (ANAT) Semantics for the Session Description Protocol (SDP) Grouping Framework*
RFC 4092	*Usage of the Session Description Protocol (SDP) Alternative Network Address Types (ANAT) Semantics in the Session Initiation Protocol (SIP)*
RFC 4109	Algorithms for Internet Key Exchange version 1 (IKEv1)
RFC 4191	*Default Router Preferences and More-Specific Routes*
RFC 4193	*Unique Local IPv6 Unicast Addresses*
RFC 4214	*Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)*
RFC 4242	*Information Refresh Time Option for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*
RFC 4282	*The Network Access Identifier*
RFC 4283	*Mobile Node Identifier Option for Mobile IPv6*
RFC 4285	*Authentication Protocol for Mobile IPv6*
RFC 4291	*IP Version 6 Addressing Architecture*
RFC 4292	*IP Forwarding Table MIB*
RFC 4293	*Management Information Base for the Internet Protocol (IP)*
RFC 4302	IP Authentication Header

RFCs	Title
RFC 4306	*Internet Key Exchange (IKEv2) Protocol*
RFC 4308	*Cryptographic Suites for IPsec*
RFC 4364	BGP MPLS/IP Virtual Private Networks (VPNs)
RFC 4382	*MPLS/BGP Layer 3 Virtual Private Network (VPN) Management Information Base*
RFC 4443	Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification
RFC 4552	*Authentication/Confidentiality for OSPFv3*
RFC 4594	*Configuration Guidelines for DiffServ Service Classes*
RFC 4601	*Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification*
RFC 4610	*Anycast-RP Using Protocol Independent Multicast (PIM)*
RFC 4649	*Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Relay Agent Remote-ID Option*
RFC 4659	BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN
RFC 4724	*Graceful Restart Mechanism for BGP*
RFC 4798	*Connecting IPv6 Islands over IPv4 MPLS Using IPv6 Provider Edge Routers (6PE)*
RFC 4861	*Neighbor Discovery for IP version 6 (IPv6)*
RFC 4862	*IPv6 Stateless Address Autoconfiguration*
RFC 4884	*Extended ICMP to Support Multi-Part Messages*
RFC 4885	*Network Mobility Support Terminology*
RFC 4887	*Network Mobility Home Network Models*
RFC 5059	*Bootstrap Router (BSR) Mechanism for Protocol Independent Multicast (PIM)*
RFC 5072	*IPv6 over PPP*
RFC 5095	*Deprecation of Type 0 Routing Headers in IPv6*
RFC 5120	*M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)*

RFCs	Title
RFC 5187	*OSPFv3 Graceful Restart*
RFC 5308	*Routing IPv6 with IS-IS*
RFC 5340	*OSPF for IPv6*
RFC 5460	*DHCPv6 Bulk Leasequery*
RFC 5838	*Support of Address Families in OSPFv3*
RFC 5881	*Bidirectional Forwarding Detection (BFD) for IPv4 and IPv6 (Single Hop)*
RFC 5905	*Network Time Protocol Version 4: Protocol and Algorithms Specification*
RFC 5969	IPv6 Rapid Deployment on IPv4 Infrastructures (6RD) -- Protocol Specification

MIBs

MIBs	MIBs Link
• CISCO-CONFIG-COPY-MIB • CISCO-CONFIG-MAN-MIB • CISCO-DATA-COLLECTION-MIB • CISCO-FLASH-MIB • CISCO-IETF-IP-FORWARDING-MIB (not available as of Cisco IOS Release 12.2(33)SRC) • CISCO-IETF-IP-MIB (not available as of Cisco IOS Release 12.2(33)SRC) • CISCO-IP-FORWARD-MIB • CISCO-IP-MIB • CISCO-RTTMON-IPv6-MIB • CISCO-SNMP-TARGET-EXT-MIB • ENTITY-MIB • NOTIFICATION-LOG-MIB • SNMP-TARGET-MIB • TUNNEL-MIB	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

Implementing IPv6 Addressing and Basic Connectivity

Implementing basic IPv6 connectivity in the Cisco IOS software consists of assigning IPv6 addresses to individual router interfaces. The forwarding of IPv6 traffic can be enabled globally, and Cisco Express Forwarding switching for IPv6 can also be enabled. Basic connectivity can be enhanced by configuring support for AAAA record types in the Domain Name System (DNS) name-to-address and address-to-name lookup processes, and by managing IPv6 neighbor discovery.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Implementing IPv6 Addressing and Basic Connectivity

- The following prerequisites apply to Cisco Express Forwarding and distributed Cisco Express Forwarding for IPv6:
    - To forward IPv6 traffic using Cisco Express Forwarding or distributed Cisco Express Forwarding, you must configure forwarding of IPv6 unicast datagrams globally on the router by using the **ipv6**

unicast-routing command, and you must configure an IPv6 address on an interface by using the **ipv6 address** command.

- You must enable Cisco Express Forwarding for IPv4 globally on the router by using the **ip cef** command before enabling Cisco Express Forwarding for IPv6 globally on the router by using the **ipv6 cef** command.
- On distributed architecture platforms that support both Cisco Express Forwarding and distributed Cisco Express Forwarding, you must enable distributed Cisco Express Forwarding for IPv4 globally on the router by using the **ip cef distributed** command before enabling distributed Cisco Express Forwarding for IPv6 globally on the router by using the **ipv6 cef distributed** command.
- To use Unicast Reverse Path Forwarding (RPF), enable Cisco Express Forwarding switching or distributed Cisco Express Forwarding switching in the router. There is no need to configure the input interface for Cisco Express Forwarding switching. As long as Cisco Express Forwarding is running on the router, individual interfaces can be configured with other switching modes.

Note For Unicast RPF to work, Cisco Express Forwarding must be configured globally in the router. Unicast RPF will not work without Cisco Express Forwarding.

Restrictions for Implementing IPv6 Addressing and Basic Connectivity

- In Cisco IOS Release 12.2(11)T or earlier releases, IPv6 supports only process switching for packet forwarding. Cisco Express Forwarding switching and distributed Cisco Express Forwarding switching for IPv6 are supported in Cisco IOS Release 12.2(13)T. Distributed Cisco Express Forwarding switching for IPv6 is supported in Cisco IOS Release 12.0(21)ST.
- IPv6 packets are transparent to Layer 2 LAN switches because the switches do not examine Layer 3 packet information before forwarding IPv6 frames. Therefore, IPv6 hosts can be directly attached to Layer 2 LAN switches.
- In any Cisco IOS release with IPv6 support, multiple IPv6 global addresses within the same prefix can be configured on an interface. However, multiple IPv6 link-local addresses on an interface are not supported. See the "Mapping IPv6 Addresses to IPv6 ATM and Frame Relay Interfaces" section for information on configuring multiple IPv6 global addresses within the same prefix on an interface.
- Because RFC 3879 deprecates the use of site-local addresses, configuration of private IPv6 addresses should be done following the recommendations of unique local addressing (ULA) in RFC 4193.
- Bridge-Group Virtual Interfaces (BVIs) in IPv6 are not supported with NAT-PT and wireless interfaces Dot11Radio.

Information About Implementing IPv6 Addressing and Basic Connectivity

- IPv6 for Cisco Software, page 41
- Large IPv6 Address Space for Unique Addresses, page 41
- IPv6 Address Formats, page 42

IPv6 for Cisco Software

IPv6, formerly named IPng (next generation), is the latest version of the Internet Protocol (IP). IP is a packet-based protocol used to exchange data, voice, and video traffic over digital networks. IPv6 was proposed when the 32-bit addressing scheme of IP version 4 (IPv4) proved to be inadequate to meet the demands of Internet growth. IPv6 is based on IP but with a much larger address space and improvements such as a simplified main header and extension headers. IPv6 is described in RFC 2460, *Internet Protocol, Version 6 (IPv6) Specification*, issued by the Internet Engineering Task Force (IETF). Further RFCs describe the architecture and services supported by IPv6.

The architecture of IPv6 has been designed to allow existing IPv4 users to transition easily to IPv6 while providing services such as end-to-end security, quality of service (QoS), and globally unique addresses. The larger IPv6 address space allows networks to scale and provide global reachability. The simplified IPv6 packet header format handles packets more efficiently. IPv6 prefix aggregation, simplified network renumbering, and IPv6 site multihoming capabilities provide an IPv6 addressing hierarchy that allows for more efficient routing. IPv6 supports widely deployed routing protocols such as Routing Information Protocol (RIP), Integrated Intermediate System-to-Intermediate System (IS-IS), Open Shortest Path First version 3 (OSPFv3), and multiprotocol Border Gateway Protocol (BGP). Other available features include stateless autoconfiguration, enhanced support for Mobile IPv6, and an increased number of multicast addresses.

Large IPv6 Address Space for Unique Addresses

The primary motivation for IPv6 is the need to meet the demand for globally unique IP addresses. IPv6 quadruples the number of network address bits from 32 bits (in IPv4) to 128 bits, which provides more than enough globally unique IP addresses for every networked device on the planet. By being globally unique, IPv6 addresses inherently enable global reachability and end-to-end security for networked devices, functionality that is crucial to the applications and services that are driving the demand for the addresses. Additionally, the flexibility of the IPv6 address space reduces the need for private addresses; therefore, IPv6 enables new application protocols that do not require special processing by border devices at the edge of networks.

IPv6 Address Formats

IPv6 addresses are represented as a series of 16-bit hexadecimal fields separated by colons (:) in the format: x:x:x:x:x:x:x:x. Following are two examples of IPv6 addresses:

2001:DB8:7654:3210:FEDC:BA98:7654:3210

2001:DB8:0:0:8:800:200C:417A

IPv6 addresses commonly contain successive hexadecimal fields of zeros. Two colons (::) may be used to compress successive hexadecimal fields of zeros at the beginning, middle, or end of an IPv6 address (the colons represent successive hexadecimal fields of zeros). The table below lists compressed IPv6 address formats.

A double colon may be used as part of the *ipv6-address* argument when consecutive 16-bit values are denoted as zero. You can configure multiple IPv6 addresses per interfaces, but only one link-local address.

Note Two colons (::) can be used only once in an IPv6 address to represent the longest successive hexadecimal fields of zeros. The hexadecimal letters in IPv6 addresses are not case-sensitive.

Table 4 Compressed IPv6 Address Formats

IPv6 Address Type	Preferred Format	Compressed Format
Unicast	2001:0:0:0:DB8:800:200C:417A	2001::DB8:800:200C:417A
Multicast	FF01:0:0:0:0:0:0:101	FF01::101
Loopback	0:0:0:0:0:0:0:1	::1
Unspecified	0:0:0:0:0:0:0:0	::

The loopback address listed in the table above may be used by a node to send an IPv6 packet to itself. The loopback address in IPv6 functions the same as the loopback address in IPv4 (127.0.0.1).

Note The IPv6 loopback address cannot be assigned to a physical interface. A packet that has the IPv6 loopback address as its source or destination address must remain within the node that created the packet. IPv6 devices do not forward packets that have the IPv6 loopback address as their source or destination address.

The unspecified address listed in the table above indicates the absence of an IPv6 address. For example, a newly initialized node on an IPv6 network may use the unspecified address as the source address in its packets until it receives its IPv6 address.

Note The IPv6 unspecified address cannot be assigned to an interface. The unspecified IPv6 addresses must not be used as destination addresses in IPv6 packets or the IPv6 routing header.

An IPv6 address prefix, in the format *ipv6-prefix*/*prefix-length*, can be used to represent bit-wise contiguous blocks of the entire address space. The *ipv6-prefix* must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons. The prefix length is

a decimal value that indicates how many of the high-order contiguous bits of the address comprise the prefix (the network portion of the address). For example, 2001:DB8:8086:6502::/32 is a valid IPv6 prefix.

IPv6 Address Type: Unicast

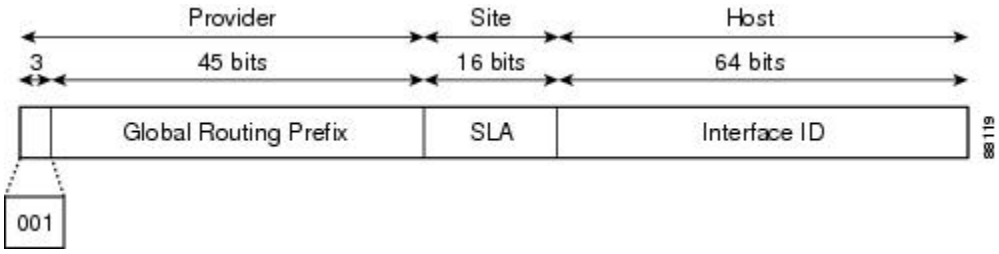
An IPv6 unicast address is an identifier for a single interface, on a single node. A packet that is sent to a unicast address is delivered to the interface identified by that address. Cisco software supports the IPv6 unicast address types described in the following sections.

Aggregatable Global Address

An aggregatable global address is an IPv6 address from the aggregatable global unicast prefix. The structure of aggregatable global unicast addresses enables strict aggregation of routing prefixes that limits the number of routing table entries in the global routing table. Aggregatable global addresses are used on links that are aggregated upward through organizations, and eventually to the ISPs.

Aggregatable global IPv6 addresses are defined by a global routing prefix, a subnet ID, and an interface ID. Except for addresses that start with binary 000, all global unicast addresses have a 64-bit interface ID. The IPv6 global unicast address allocation uses the range of addresses that start with binary value 001 (2000::/3). The figure below shows the structure of an aggregatable global address.

Figure 1 Aggregatable Global Address Format



Addresses with a prefix of 2000::/3 (001) through E000::/3 (111) are required to have 64-bit interface identifiers in the extended universal identifier (EUI)-64 format. The Internet Assigned Numbers Authority (IANA) allocates the IPv6 address space in the range of 2000::/16 to regional registries.

The aggregatable global address typically consists of a 48-bit global routing prefix and a 16-bit subnet ID or site-level aggregator (SLA). In the IPv6 aggregatable global unicast address format document (RFC 2374), the global routing prefix included two other hierarchically structured fields named top-level aggregator (TLA) and next-level aggregator (NLA). The Internet Engineering Task Force (IETF) decided to remove the TLA and NLA fields from the RFCs because these fields are policy-based. Some existing IPv6 networks deployed before the change might still be using networks based on the older architecture.

A 16-bit subnet field called the subnet ID could be used by individual organizations to create their own local addressing hierarchy and to identify subnets. A subnet ID is similar to a subnet in IPv4, except that an organization with an IPv6 subnet ID can support up to 65,535 individual subnets.

An interface ID is used to identify interfaces on a link. The interface ID must be unique to the link. It may also be unique over a broader scope. In many cases, an interface ID will be the same as or based on the

link-layer address of an interface. Interface IDs used in aggregatable global unicast and other IPv6 address types must be 64 bits long and constructed in the modified EUI-64 format.
Interface IDs are constructed in the modified EUI-64 format in one of the following ways:

- For all IEEE 802 interface types (for example, Ethernet and FDDI interfaces), the first three octets (24 bits) are taken from the Organizationally Unique Identifier (OUI) of the 48-bit link-layer address (the media access control, or MAC, address) of the interface, the fourth and fifth octets (16 bits) are a fixed hexadecimal value of FFFE, and the last three octets (24 bits) are taken from the last three octets of the MAC address. The construction of the interface ID is completed by setting the universal/local (U/L) bit--the seventh bit of the first octet--to a value of 0 or 1. A value of 0 indicates a locally administered identifier; a value of 1 indicates a globally unique IPv6 interface identifier.
- For other interface types (for example, ATM, Frame Relay, loopback, serial, and tunnel interface types except tunnel interfaces used with IPv6 overlay tunnels), the interface ID is constructed in the same way as the interface ID for IEEE 802 interface types; however, the first MAC address from the pool of MAC addresses in the device is used to construct the identifier (because the interface does not have a MAC address).
- For tunnel interface types that are used with IPv6 overlay tunnels, the interface ID is the IPv4 address assigned to the tunnel interface with all zeros in the high-order 32 bits of the identifier.

Note For interfaces using point-to-point protocol (PPP), given that the interfaces at both ends of the connection might have the same MAC address, the interface identifiers used at both ends of the connection are negotiated (picked randomly and, if necessary, reconstructed) until both identifiers are unique. The first MAC address in the device is used to construct the identifier for interfaces using PPP.

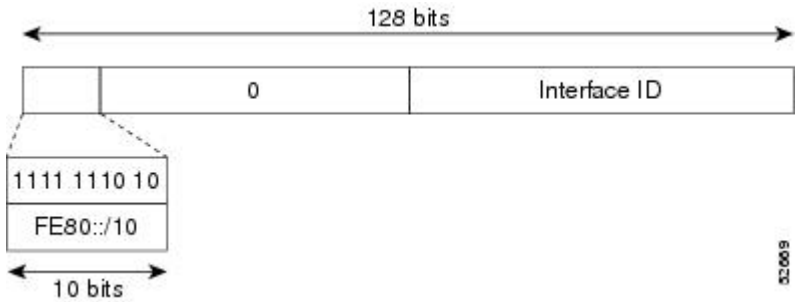
If no IEEE 802 interface types are in the device, link-local IPv6 addresses are generated on the interfaces in the device in the following sequence:

1. The device is queried for MAC addresses (from the pool of MAC addresses in the device).
2. If no MAC addresses are available in the device, the serial number of the device is used to form the link-local addresses.
3. If the serial number of the device cannot be used to form the link-local addresses, the device uses a message digest algorithm 5 (MD5) hash to determine the MAC address of the device from the hostname of the device.

Link-Local Address

A link-local address is an IPv6 unicast address that can be automatically configured on any interface using the link-local prefix FE80::/10 (1111 1110 10) and the interface identifier in the modified EUI-64 format. Link-local addresses are used in the neighbor discovery protocol and the stateless autoconfiguration process. Nodes on a local link can use link-local addresses to communicate; the nodes do not need globally unique addresses to communicate. The figure below shows the structure of a link-local address.
IPv6 devices must not forward packets that have link-local source or destination addresses to other links.

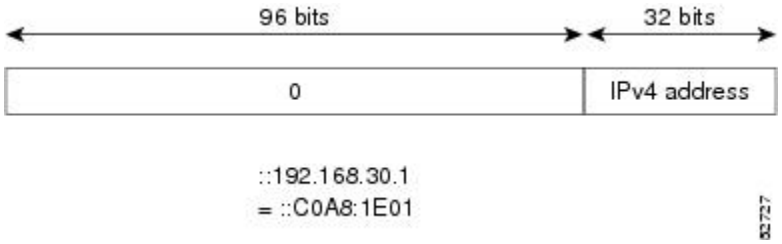
Figure 2 Link-Local Address Format



IPv4-Compatible IPv6 Address

An IPv4-compatible IPv6 address is an IPv6 unicast address that has zeros in the high-order 96 bits of the address and an IPv4 address in the low-order 32 bits of the address. The format of an IPv4-compatible IPv6 address is 0:0:0:0:0:0:A.B.C.D or ::A.B.C.D. The entire 128-bit IPv4-compatible IPv6 address is used as the IPv6 address of a node and the IPv4 address embedded in the low-order 32 bits is used as the IPv4 address of the node. IPv4-compatible IPv6 addresses are assigned to nodes that support both the IPv4 and IPv6 protocol stacks and are used in automatic tunnels. The figure below shows the structure of an IPv4-compatible IPv6 address and a few acceptable formats for the address.

Figure 3 IPv4-Compatible IPv6 Address Format



Unique Local Address

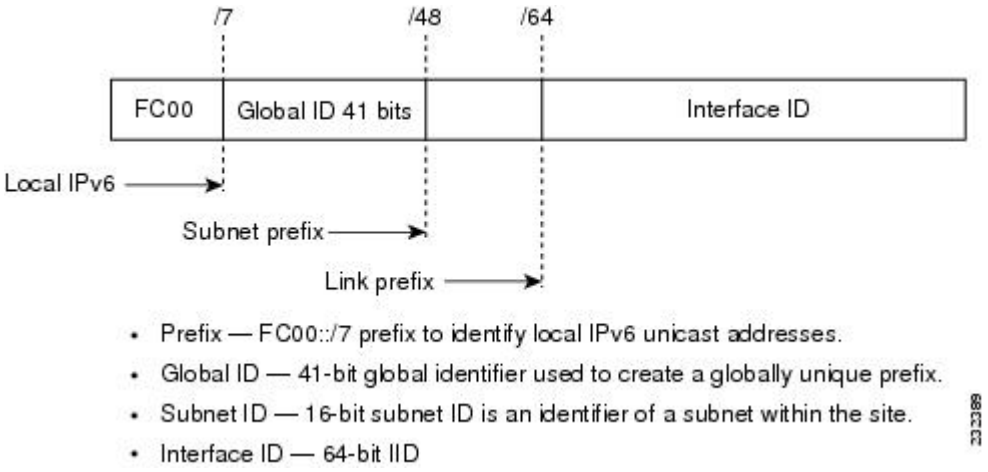
A unique local address is an IPv6 unicast address that is globally unique and is intended for local communications. It is not expected to be routable on the global Internet and is routable inside of a limited area, such as a site. It may also be routed between a limited set of sites.

A unique local address has the following characteristics:

- It has a globally unique prefix (that is, it has a high probability of uniqueness).
- It has a well-known prefix to allow for easy filtering at site boundaries.
- It allows sites to be combined or privately interconnected without creating any address conflicts or requiring renumbering of interfaces that use these prefixes.
- It is ISP-independent and can be used for communications inside of a site without having any permanent or intermittent Internet connectivity.
- If it is accidentally leaked outside of a site via routing or DNS, there is no conflict with any other addresses.
- Applications may treat unique local addresses like global scoped addresses.

The figure below shows the structure of a unique local address.

Figure 4 Unique Local Address Structure



Site-Local Address

Because RFC 3879 obsoletes the use of site-local addresses, configuration of private IPv6 addresses should be done following the recommendations of unique local addressing in RFC 4193.

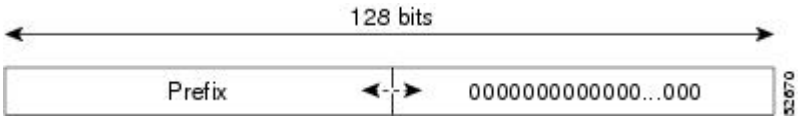
IPv6 Address Type: Anycast

An anycast address is an address that is assigned to a set of interfaces that typically belong to different nodes. A packet sent to an anycast address is delivered to the closest interface (as defined by the routing protocols in use) identified by the anycast address. Anycast addresses are syntactically indistinguishable from unicast addresses, because anycast addresses are allocated from the unicast address space. Assigning a unicast address to more than one interface makes a unicast address an anycast address. Nodes to which the anycast address is assigned must be explicitly configured to recognize that the address is an anycast address.

Note Anycast addresses can be used only by a device, not a host, and anycast addresses must not be used as the source address of an IPv6 packet.

The figure below shows the format of the subnet device anycast address; the address has a prefix concatenated by a series of zeros (the interface ID). The subnet device anycast address can be used to reach a device on the link that is identified by the prefix in the subnet device anycast address.

Figure 5 Subnet Device Anycast Address Format

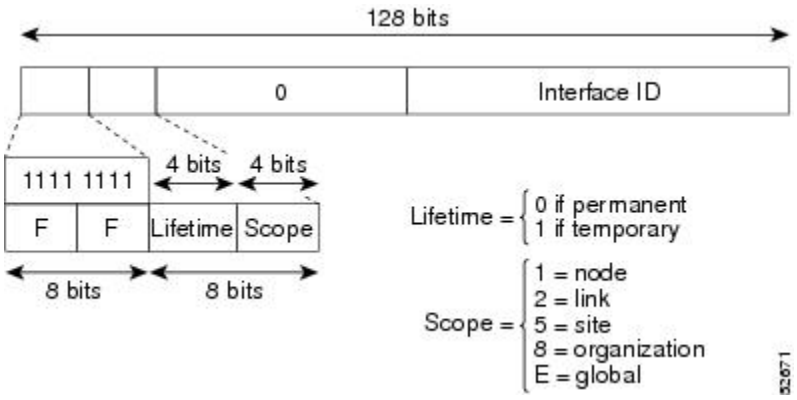


IPv6 Address Type Multicast

An IPv6 multicast address is an IPv6 address that has a prefix of FF00::/8 (1111 1111). An IPv6 multicast address is an identifier for a set of interfaces that typically belong to different nodes. A packet sent to a multicast address is delivered to all interfaces identified by the multicast address. The second octet following the prefix defines the lifetime and scope of the multicast address. A permanent multicast address has a lifetime parameter equal to 0; a temporary multicast address has a lifetime parameter equal to 1. A multicast address that has the scope of a node, link, site, or organization, or a global scope has a scope parameter of 1, 2, 5, 8, or E, respectively. For example, a multicast address with the prefix FF02::/16 is a

permanent multicast address with a link scope. The figure below shows the format of the IPv6 multicast address.

Figure 6 IPv6 Multicast Address Format



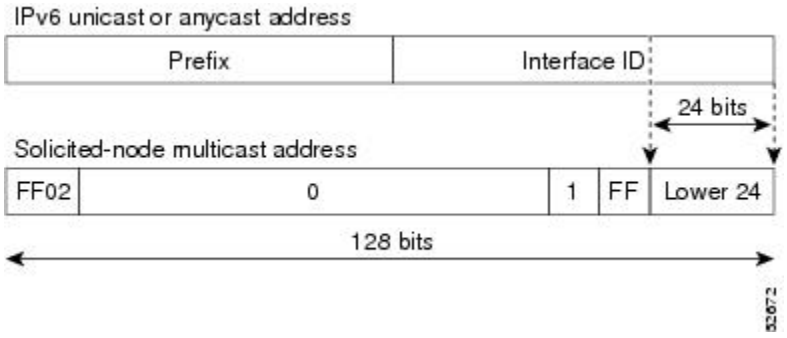
IPv6 nodes (hosts and routers) are required to join (receive packets destined for) the following multicast groups:

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

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

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

Figure 7 IPv6 Solicited-Node Multicast Address Format



Note There are no broadcast addresses in IPv6. IPv6 multicast addresses are used instead of broadcast addresses.

- IPv6 Multicast Groups, page 48

IPv6 Multicast Groups

An IPv6 address must be configured on an interface before the interface can forward IPv6 traffic. Configuring a site-local or global IPv6 address on an interface automatically configures a link-local address and activates IPv6 for that interface. Additionally, the configured interface automatically joins the following required multicast groups for that link:

- Solicited-node multicast group FF02:0:0:0:0:1:FF00::/104 for each unicast and anycast address assigned to the interface

Note The solicited-node multicast address is used in the neighbor discovery process.

- All-nodes link-local multicast group FF02::1
- All-routers link-local multicast group FF02::2

IPv6 Address Output Display

When IPv6 or IPv4 command output displays an IPv6 address, a long IPv6 address can overflow into neighboring fields, causing the output to be difficult to read. The output fields were designed to work with the longest possible IPv4 address, which has 15 characters; IPv6 addresses can be up to 39 characters long. The following scheme has been adopted in IPv4 and IPv6 commands to allow the appropriate length of IPv6 address to be displayed and move the following fields to the next line, if necessary. The fields that are moved are kept in alignment with the header row.

The following example displays eight connections. The first six connections feature IPv6 addresses; the last two connections feature IPv4 addresses.

```
Device# where
Conn Host                Address             Byte  Idle Conn Name
   1 test5               2001:DB8:3333:4::5    6    24 test5
   2 test4               2001:DB8:3333:44::5
                                               6    24 test4
   3 2001:DB8:3333:4::5 2001:DB8:3333:4::5     6    24 2001:DB8:3333:4::5
   4 2001:DB8:3333:44::5
                         2001:DB8:3333:44::5
                                               6    23 2001:DB8:3333:44::5
   5 2001:DB8:3000:4000:5000:6000:7000:8001
                         2001:DB8:3000:4000:5000:6000:7000:8001
                                               6    20 2001:DB8:3000:4000:5000:6000:
   6 2001:DB8:1::1      2001:DB8:1::1          0     1 2001:DB8:1::1
   7 10.1.9.1            10.1.9.1              0     0 10.1.9.1
   8 10.222.111.222      10.222.111.222        0     0 10.222.111.222
```

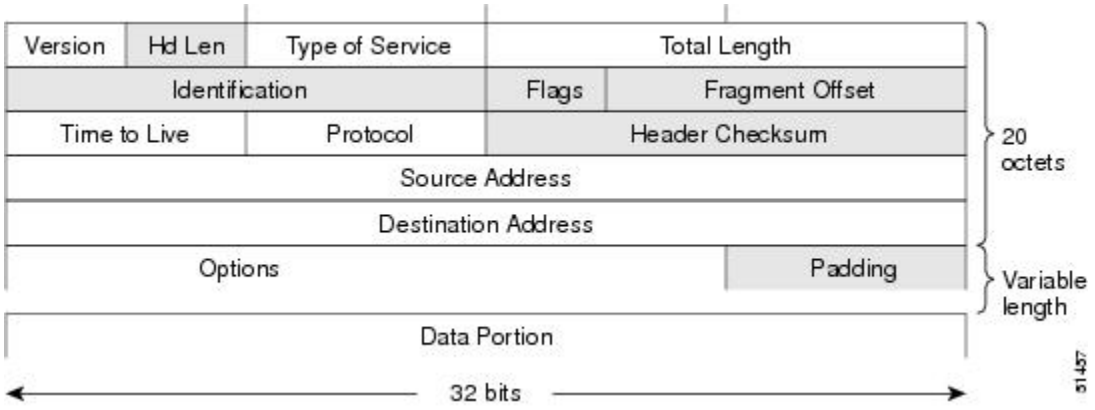
Connection 1 contains an IPv6 address that uses the maximum address length in the address field. Connection 2 shows the IPv6 address overflowing the address field and the following fields moved to the next line, but in alignment with the appropriate headers. Connection 3 contains an IPv6 address that fills the maximum length of the hostname and address fields without wrapping any lines. Connection 4 shows the effect of both the hostname and address fields containing a long IPv6 address. The output is shown over three lines keeping the correct heading alignment. Connection 5 displays a similar effect as connection 4 with a very long IPv6 address in the hostname and address fields. Note that the connection name field is actually truncated. Connection 6 displays a very short IPv6 address that does not require any change in the display. Connections 7 and 8 display short and long IPv4 addresses.

Note The IPv6 address output display applies to all commands that display IPv6 addresses.

Simplified IPv6 Packet Header

The basic IPv4 packet header has 12 fields with a total size of 20 octets (160 bits) (see the figure below). The 12 fields may be followed by an Options field, which is followed by a data portion that is usually the transport-layer packet. The variable length of the Options field adds to the total size of the IPv4 packet header. The shaded fields of the IPv4 packet header shown in the figure below are not included in the IPv6 packet header.

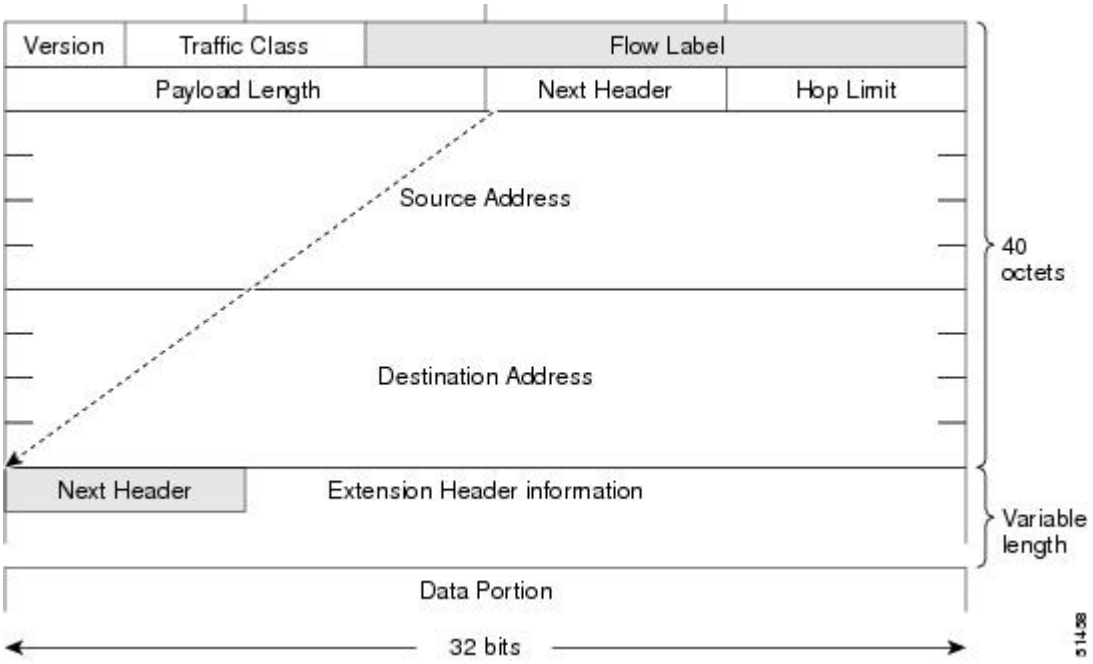
Figure 8 IPv4 Packet Header Format



The basic IPv6 packet header has 8 fields with a total size of 40 octets (320 bits) (see the figure below). Fields were removed from the IPv6 header because, in IPv6, fragmentation is not handled by devices and checksums at the network layer are not used. Instead, fragmentation in IPv6 is handled by the source of a packet and checksums at the data link layer and transport layer are used. (In IPv4, the UDP transport layer uses an optional checksum. In IPv6, use of the UDP checksum is required to check the integrity of the inner

packet.) Additionally, the basic IPv6 packet header and Options field are aligned to 64 bits, which can facilitate the processing of IPv6 packets.

Figure 9 IPv6 Packet Header Format



The table below lists the fields in the basic IPv6 packet header.

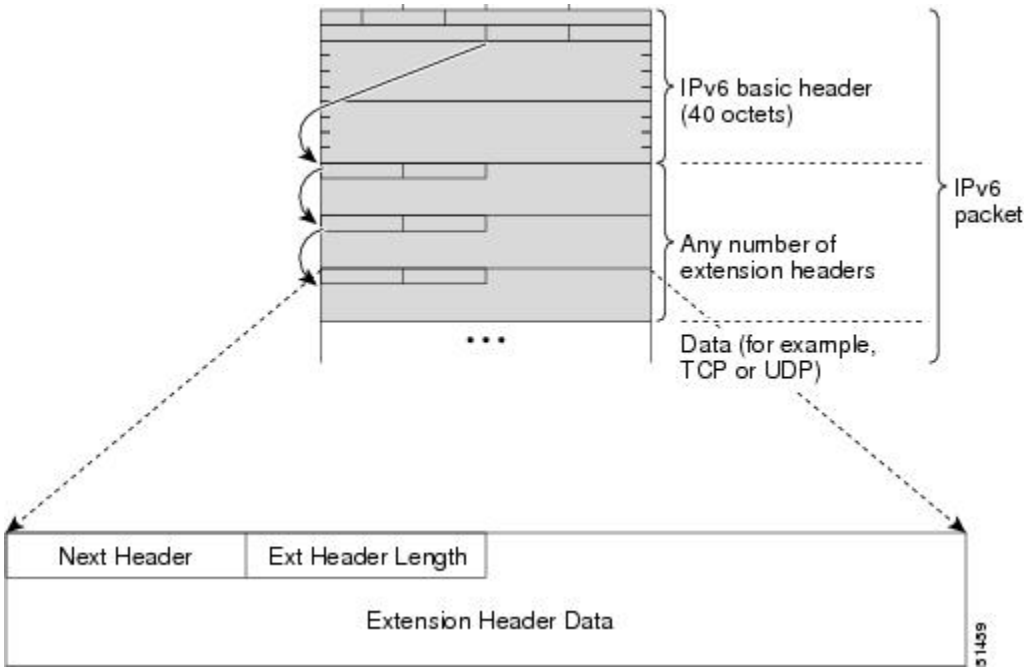
Table 5 Basic IPv6 Packet Header Fields

Field	Description
Version	Similar to the Version field in the IPv4 packet header, except that the field lists number 6 for IPv6 instead of number 4 for IPv4.
Traffic Class	Similar to the Type of Service field in the IPv4 packet header. The Traffic Class field tags packets with a traffic class that is used in differentiated services.
Flow Label	A new field in the IPv6 packet header. The Flow Label field tags packets with a specific flow that differentiates the packets at the network layer.
Payload Length	Similar to the Total Length field in the IPv4 packet header. The Payload Length field indicates the total length of the data portion of the packet.

Field	Description
Next Header	Similar to the Protocol field in the IPv4 packet header. The value of the Next Header field determines the type of information following the basic IPv6 header. The type of information following the basic IPv6 header can be a transport-layer packet, for example, a TCP or UDP packet, or an Extension Header, as shown in the figure immediately above.
Hop Limit	Similar to the Time to Live field in the IPv4 packet header. The value of the Hop Limit field specifies the maximum number of devices that an IPv6 packet can pass through before the packet is considered invalid. Each device decrements the value by one. Because no checksum is in the IPv6 header, the device can decrement the value without needing to recalculate the checksum, which saves processing resources.
Source Address	Similar to the Source Address field in the IPv4 packet header, except that the field contains a 128-bit source address for IPv6 instead of a 32-bit source address for IPv4.
Destination Address	Similar to the Destination Address field in the IPv4 packet header, except that the field contains a 128-bit destination address for IPv6 instead of a 32-bit destination address for IPv4.

Following the eight fields of the basic IPv6 packet header are optional extension headers and the data portion of the packet. If present, each extension header is aligned to 64 bits. There is no fixed number of extension headers in an IPv6 packet. The extension headers form a chain of headers. Each extension header is identified by the Next Header field of the previous header. Typically, the final extension header has a Next Header field of a transport-layer protocol, such as TCP or UDP. The figure below shows the IPv6 extension header format.

Figure 10 IPv6 Extension Header Format



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

Table 6 IPv6 Extension Header Types

Header Type	Next Header Value	Description
Hop-by-hop options header	0	This header is processed by all hops in the path of a packet. When present, the hop-by-hop options header always follows immediately after the basic IPv6 packet header.
Destination options header	60	The destination options header can follow any hop-by-hop options header, in which case the destination options header is processed at the final destination and also at each visited address specified by a routing header. Alternatively, the destination options header can follow any Encapsulating Security Payload (ESP) header, in which case the destination options header is processed only at the final destination.
Routing header	43	The routing header is used for source routing.
Fragment header	44	The fragment header is used when a source must fragment a packet that is larger than the maximum transmission unit (MTU) for the path between itself and a destination. The Fragment header is used in each fragmented packet.
Authentication header and ESP header	51 50	The Authentication header and the ESP header are used within IP Security Protocol (IPsec) to provide authentication, integrity, and confidentiality of a packet. These headers are identical for both IPv4 and IPv6.
Upper-layer headers	6 (TCP) 17 (UDP)	The upper-layer (transport) headers are the typical headers used inside a packet to transport the data. The two main transport protocols are TCP and UDP.
Mobility headers	135	Extension headers used by mobile nodes, correspondent nodes, and home agents in all messaging related to the creation and management of bindings.

Cisco Express Forwarding and Distributed Cisco Express Forwarding Switching for IPv6

Cisco Express Forwarding is advanced, Layer 3 IP switching technology for the forwarding of IPv6 packets. Distributed Cisco Express Forwarding performs the same functions as Cisco Express Forwarding but for distributed architecture platforms. Distributed Cisco Express Forwarding for IPv6 and Cisco

Express Forwarding for IPv6 function the same and offer the same benefits as for distributed Cisco Express Forwarding for IPv4 and Cisco Express Forwarding for IPv4. Both have network entries that are added, removed, or modified in the IPv6 Routing Information Base (RIB) (as dictated by the routing protocols in use) and are reflected in the Forwarding Information Bases (FIBs), and the IPv6 adjacency tables maintain Layer 2 next-hop addresses for all entries in each FIB.

Each IPv6 device interface has an association to one IPv6 global FIB and one IPv6 link-local FIB (multiple interfaces can have an association to the same FIB). All IPv6 device interfaces that are attached to the same IPv6 link share the same IPv6 link-local FIB. IPv6 packets that have an IPv6 global destination address are processed by the IPv6 global FIB; however, packets that have an IPv6 global destination address and an IPv6 link-local source address are sent to the Route Processor (RP) for process switching and scope-error handling. Packets that have a link-local source address are not forwarded off of the local link and are sent to the RP for process switching and scope-error handling.

- Unicast Reverse Path Forwarding, page 53

Unicast Reverse Path Forwarding

Use the Unicast Reverse Path Forwarding for IPv6 feature to mitigate problems caused by malformed or spoofed IPv6 source addresses that pass through an IPv6 device. Malformed or forged source addresses can indicate denial-of-service (DoS) attacks based on source IPv6 address spoofing.

When uRPF is enabled on an interface, the device examines all packets received on that interface. The device verifies that the source address appears in the routing table and matches the interface on which the packet was received. This "look backward" ability is available only when Cisco Express Forwarding is enabled on the device, because the lookup relies on the presence of the Forwarding Information Bases (FIBs). Cisco Express Forwarding generates the FIB as part of its operation.

Note uRPF is an input function and is applied only on the input interface of a device at the upstream end of a connection.

The uRPF feature verifies whether any packet received at a device interface arrives on one of the best return paths to the source of the packet. The feature performs a reverse lookup in the Cisco Express Forwarding table. If uRPF does not find a reverse path for the packet, uRPF can drop or forward the packet, depending on whether an access control list (ACL) is specified. If an ACL is specified, then when (and only when) a packet fails the uRPF check, the ACL is checked to verify if the packet should be dropped (using a deny statement in the ACL) or forwarded (using a permit statement in the ACL). Regardless of whether a packet is dropped or forwarded, the packet is counted in the global IP traffic statistics for uRPF drops and in the interface statistics for uRPF.

If no ACL is specified, the device drops the forged or malformed packet immediately and no ACL logging occurs. The device and interface uRPF counters are updated.

uRPF events can be logged by specifying the logging option for the ACL entries. Log information can be used to gather information about the attack, such as source address and time.

Note With uRPF, all equal-cost "best" return paths are considered valid. uRPF works in cases where multiple return paths exist, provided that each path is equal to the others in terms of the routing cost (number of hops, weights, and so on) and as long as the route is in the FIB.

DNS for IPv6

IPv6 supports DNS record types that are supported in the DNS name-to-address and address-to-name lookup processes. The DNS record types support IPv6 addresses. IPv6 also supports the reverse mapping of IPv6 addresses to DNS names.

A name server is used to track information associated with domain names. A name server can maintain a database of hostname-to-address mappings. Each name can map to one or more IPv4 addresses, IPv6 addresses, or both address types. In order to use this service to map domain names to IPv6 addresses, you must specify a name server and enable the DNS.

Cisco software maintains a cache of hostname-to-address mappings for use by the **connect**, **telnet**, and **ping** commands, related Telnet support operations, and many other commands that generate command output. This cache speeds the conversion of names to addresses.

Similar to IPv4, IPv6 uses a naming scheme that allows a network device to be identified by its location within a hierarchical name space that provides for domains. Domain names are joined with periods (.) as the delimiting characters. For example, Cisco is a commercial organization that is identified by a com domain name, so its domain name is cisco.com. A specific device in this domain, the FTP server, for example, is identified as ftp.cisco.com.

The following table lists the IPv6 DNS record types.

Table 7 IPv6 DNS Record Types

Record Type	Description	Format
AAAA	Maps a hostname to an IPv6 address. (Equivalent to an A record in IPv4.)	www.abc.test AAAA 3FFE:YYYY:C18:1::2
PTR	Maps an IPv6 address to a hostname. (Equivalent to a pointer record [PTR] in IPv4.) **Note** Cisco software supports resolution of PTR records for the IP6.INT domain.	2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.1.0.0.0.8.1.c.0.y.y.y.y.e.f.f.3.ip6.int PTR www.abc.test

Path MTU Discovery for IPv6

As in IPv4, path MTU discovery in IPv6 allows a host to dynamically discover and adjust to differences in the MTU size of every link along a given data path. In IPv6, however, fragmentation is handled by the source of a packet when the path MTU of one link along a given data path is not large enough to accommodate the size of the packets. Having IPv6 hosts handle packet fragmentation saves IPv6 router processing resources and helps IPv6 networks run more efficiently.

Note In IPv6, the minimum link MTU is 1280 octets. Cisco recommends using an MTU value of 1500 octets for IPv6 links.

Cisco Discovery Protocol IPv6 Address Support

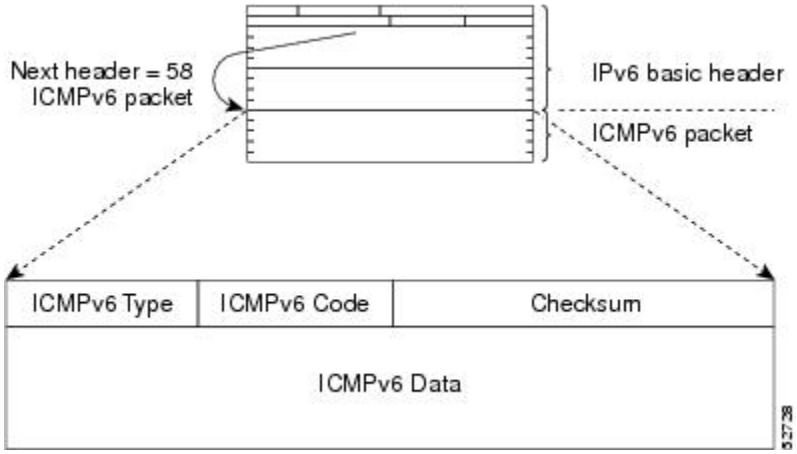
The Cisco Discovery Protocol IPv6 address support for neighbor information feature adds the ability to transfer IPv6 addressing information between two Cisco devices. Cisco Discovery Protocol support for IPv6 addresses provides IPv6 information to network management products and troubleshooting tools.

ICMP for IPv6

Internet Control Message Protocol (ICMP) in IPv6 functions the same as ICMP in IPv4. ICMP generates error messages, such as ICMP destination unreachable messages, and informational messages, such as ICMP echo request and reply messages. Additionally, ICMP packets in IPv6 are used in the IPv6 neighbor discovery process, path MTU discovery, and the Multicast Listener Discovery (MLD) protocol for IPv6. MLD is used by IPv6 devices to discover multicast listeners (nodes that want to receive multicast packets destined for specific multicast addresses) on directly attached links. MLD is based on version 2 of the Internet Group Management Protocol (IGMP) for IPv4.

A value of 58 in the Next Header field of the basic IPv6 packet header identifies an IPv6 ICMP packet. ICMP packets in IPv6 are like a transport-layer packet in the sense that the ICMP packet follows all the extension headers and is the last piece of information in the IPv6 packet. Within IPv6 ICMP packets, the ICMPv6 Type and ICMPv6 Code fields identify IPv6 ICMP packet specifics, such as the ICMP message type. The value in the Checksum field is derived (computed by the sender and checked by the receiver) from the fields in the IPv6 ICMP packet and the IPv6 pseudoheader. The ICMPv6 Data field contains error or diagnostic information relevant to IP packet processing. The figure below shows the IPv6 ICMP packet header format.

Figure 11 IPv6 ICMP Packet Header Format



- IPv6 ICMP Rate Limiting, page 55

IPv6 ICMP Rate Limiting

The IPv6 ICMP rate limiting feature implements a token bucket algorithm for limiting the rate at which IPv6 ICMP error messages are sent out on the network. The initial implementation of IPv6 ICMP rate limiting defined a fixed interval between error messages, but some applications such as traceroute often require replies to a group of requests sent in rapid succession. The fixed interval between error messages is not flexible enough to work with applications such as traceroute and can cause the application to fail.

Implementing a token bucket scheme allows a number of tokens--representing the ability to send one error message each--to be stored in a virtual bucket. The maximum number of tokens allowed in the bucket can be specified, and for every error message to be sent, one token is removed from the bucket. If a series of error messages is generated, error messages can be sent until the bucket is empty. When the bucket is empty of tokens, no IPv6 ICMP error messages are sent until a new token is placed in the bucket. The token bucket algorithm does not increase the average rate limiting time interval, and it is more flexible than the fixed time interval scheme.

IPv6 Neighbor Discovery

The IPv6 neighbor discovery process uses ICMP messages and solicited-node multicast addresses to determine the link-layer address of a neighbor on the same network (local link), verify the reachability of a neighbor, and track neighboring devices.

The IPv6 static cache entry for neighbor discovery feature allows static entries to be made in the IPv6 neighbor cache. Static routing requires an administrator to manually enter IPv6 addresses, subnet masks, gateways, and corresponding Media Access Control (MAC) addresses for each interface of each device into a table. Static routing enables more control but requires more work to maintain the table. The table must be updated each time routes are added or changed.

Stateful Switchover

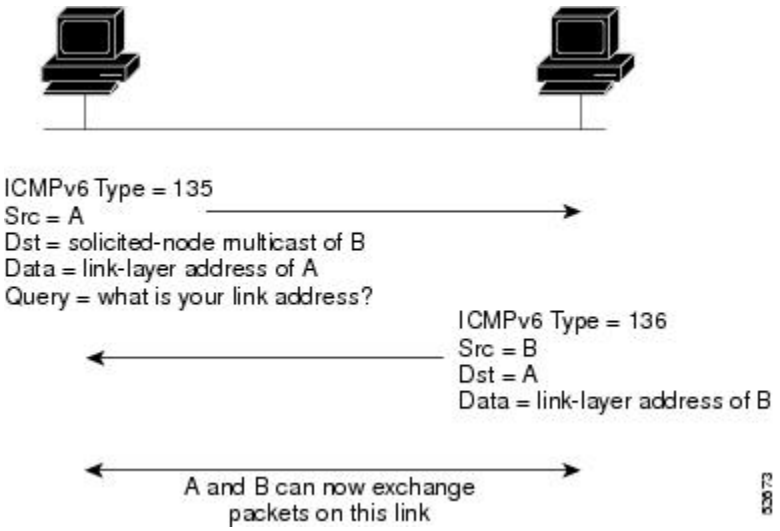
IPv6 neighbor discovery supports stateful switchover (SSO) using Cisco Express Forwarding. When switchover occurs, the Cisco Express Forwarding adjacency state, which is checkpointed, is used to reconstruct the neighbor discovery cache.

IPv6 Neighbor Solicitation Message

A value of 135 in the Type field of the ICMP packet header identifies a neighbor solicitation message. Neighbor solicitation messages are sent on the local link when a node wants to determine the link-layer address of another node on the same local link (see the figure below). When a node wants to determine the link-layer address of another node, the source address in a neighbor solicitation message is the IPv6 address of the node sending the neighbor solicitation message. The destination address in the neighbor solicitation

message is the solicited-node multicast address that corresponds to the IPv6 address of the destination node. The neighbor solicitation message also includes the link-layer address of the source node.

Figure 12 IPv6 Neighbor Discovery: Neighbor Solicitation Message



After receiving the neighbor solicitation message, the destination node replies by sending a neighbor advertisement message, which has a value of 136 in the Type field of the ICMP packet header, on the local link. The source address in the neighbor advertisement message is the IPv6 address of the node (more specifically, the IPv6 address of the node interface) sending the neighbor advertisement message. The destination address in the neighbor advertisement message is the IPv6 address of the node that sent the neighbor solicitation message. The data portion of the neighbor advertisement message includes the link-layer address of the node sending the neighbor advertisement message.

After the source node receives the neighbor advertisement, the source node and destination node can communicate.

Neighbor solicitation messages are also used to verify the reachability of a neighbor after the link-layer address of a neighbor is identified. When a node wants to verify the reachability of a neighbor, the destination address in a neighbor solicitation message is the unicast address of the neighbor.

Neighbor advertisement messages are also sent when there is a change in the link-layer address of a node on a local link. When there is such a change, the destination address for the neighbor advertisement is the all-nodes multicast address.

Neighbor solicitation messages are also used to verify the reachability of a neighbor after the link-layer address of a neighbor is identified. Neighbor unreachability detection identifies the failure of a neighbor or the failure of the forward path to the neighbor, and is used for all paths between hosts and neighboring nodes (hosts or devices). Neighbor unreachability detection is performed for neighbors to which only unicast packets are being sent and is not performed for neighbors to which multicast packets are being sent.

A neighbor is considered reachable when a positive acknowledgment is returned from the neighbor (indicating that packets previously sent to the neighbor have been received and processed). A positive acknowledgment from an upper-layer protocol (such as TCP) indicates that a connection is making forward progress (reaching its destination) or the receipt of a neighbor advertisement message in response to a neighbor solicitation message. If packets are reaching the peer, they are also reaching the next-hop neighbor of the source. Therefore, forward progress is also a confirmation that the next-hop neighbor is reachable.

For destinations that are not on the local link, forward progress implies that the first-hop device is reachable. When acknowledgments from an upper-layer protocol are not available, a node probes the neighbor using unicast neighbor solicitation messages to verify that the forward path is still working.

The return of a solicited neighbor advertisement message from the neighbor is a positive acknowledgment that the forward path is still working (neighbor advertisement messages that have the solicited flag set to a value of 1 are sent only in response to a neighbor solicitation message). Unsolicited messages confirm only the one-way path from the source to the destination node; solicited neighbor advertisement messages indicate that a path is working in both directions.

Note A neighbor advertisement message that has the solicited flag set to a value of 0 must not be considered as a positive acknowledgment that the forward path is still working.

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

Every IPv6 unicast address (global or link-local) must be verified for uniqueness on the link; however, until the uniqueness of the link-local address is verified, duplicate address detection is not performed on any other IPv6 addresses associated with the link-local address. The Cisco implementation of duplicate address detection in the Cisco software does not verify the uniqueness of anycast or global addresses that are generated from 64-bit interface identifiers.

Enhanced IPv6 Neighbor Discovery Cache Management

The enhanced IPv6 neighbor discovery cache management feature optimizes IPv6 neighbor discovery by providing ND cache autorefresh, unsolicited neighbor advertisement (NA) gleaning, and neighbor unreachability detection (NUD) exponential retransmit.

The neighbor discovery protocol enforces NUD, which can detect failing nodes or devices and changes to link-layer addresses. NUD is used to maintain reachability information for all paths between hosts and neighboring nodes, including host-to-host, host-to-device, and device-to-host communication.

The neighbor cache maintains mapping information about the IPv6 link-local or global address to the link-layer address. The neighbor cache also maintains the neighbor's reachability state, which is updated using NUD. Neighbors can be in one of the following five possible states:

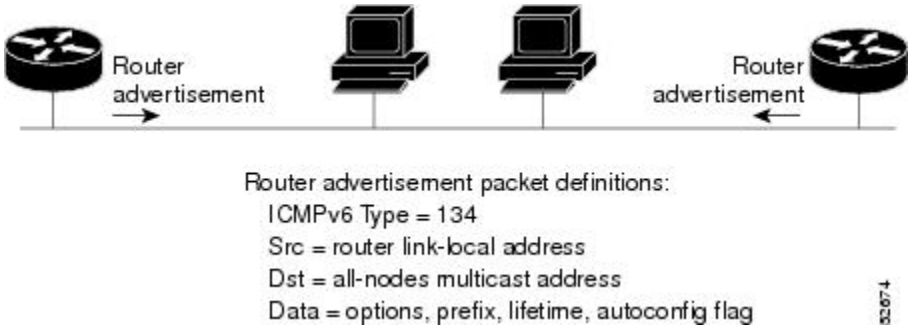
- DELAY—Neighbor is pending re-resolution, and traffic might flow to this neighbor.
- INCOMPLETE—Address resolution is in progress, and the link-layer address is not yet known.
- PROBE—Neighbor re-resolution is in progress, and traffic might flow to this neighbor.
- REACHABLE—Neighbor is known to be reachable within the last reachable time interval.
- STALE—Neighbor requires re-resolution, and traffic may flow to this neighbor.
-

IPv6 Router Advertisement Message

Router advertisement (RA) messages, which have a value of 134 in the Type field of the ICMP packet header, are periodically sent out each configured interface of an IPv6 device. For stateless autoconfiguration to work properly, the advertised prefix length in RA messages must always be 64 bits.

The RA messages are sent to the all-nodes multicast address (see the figure below).

Figure 13 IPv6 Neighbor Discovery: RA Message



RA messages typically include the following information:

- One or more onlink IPv6 prefixes that nodes on the local link can use to automatically configure their IPv6 addresses
- Lifetime information for each prefix included in the advertisement
- Sets of flags that indicate the type of autoconfiguration (stateless or stateful) that can be completed
- Default device information (whether the device sending the advertisement should be used as a default device and, if so, the amount of time, in seconds, the device should be used as a default device)
- Additional information for hosts, such as the hop limit and maximum transmission unit (MTU) a host should use in packets that it originates

RAs are also sent in response to device solicitation messages. Device solicitation messages, which have a value of 133 in the Type field of the ICMP packet header, are sent by hosts at system startup so that the host can immediately autoconfigure without needing to wait for the next scheduled RA message. Given that device solicitation messages are usually sent by hosts at system startup (the host does not have a configured unicast address), the source address in device solicitation messages is usually the unspecified IPv6 address (0:0:0:0:0:0:0:0). If the host has a configured unicast address, the unicast address of the interface sending the device solicitation message is used as the source address in the message. The destination address in device solicitation messages is the all-devices multicast address with a scope of the link. When an RA is sent in response to a device solicitation, the destination address in the RA message is the unicast address of the source of the device solicitation message.

The following RA message parameters can be configured:

- The time interval between periodic RA messages
- The “device lifetime” value, which indicates the usefulness of a device as the default device (for use by all nodes on a given link)
- The network prefixes in use on a given link
- The time interval between neighbor solicitation message retransmissions (on a given link)
- The amount of time a node considers a neighbor reachable (for use by all nodes on a given link)

The configured parameters are specific to an interface. The sending of RA messages (with default values) is automatically enabled on Ethernet and FDDI interfaces when the **ipv6 unicast-routing** command is

configured. For other interface types, the sending of RA messages must be manually configured by using the **no ipv6 nd ra suppress** command. The sending of RA messages can be disabled on individual interfaces by using the **ipv6 nd ra suppress** command.

- Default Router Preferences for Traffic Engineering, page 60

Default Router Preferences for Traffic Engineering

Hosts discover and select default devices by listening to Router Advertisements (RAs). Typical default device selection mechanisms are suboptimal in certain cases, such as when traffic engineering is needed. For example, two devices on a link may provide equivalent but not equal-cost routing, and policy may dictate that one of the devices is preferred. Some examples are as follows:

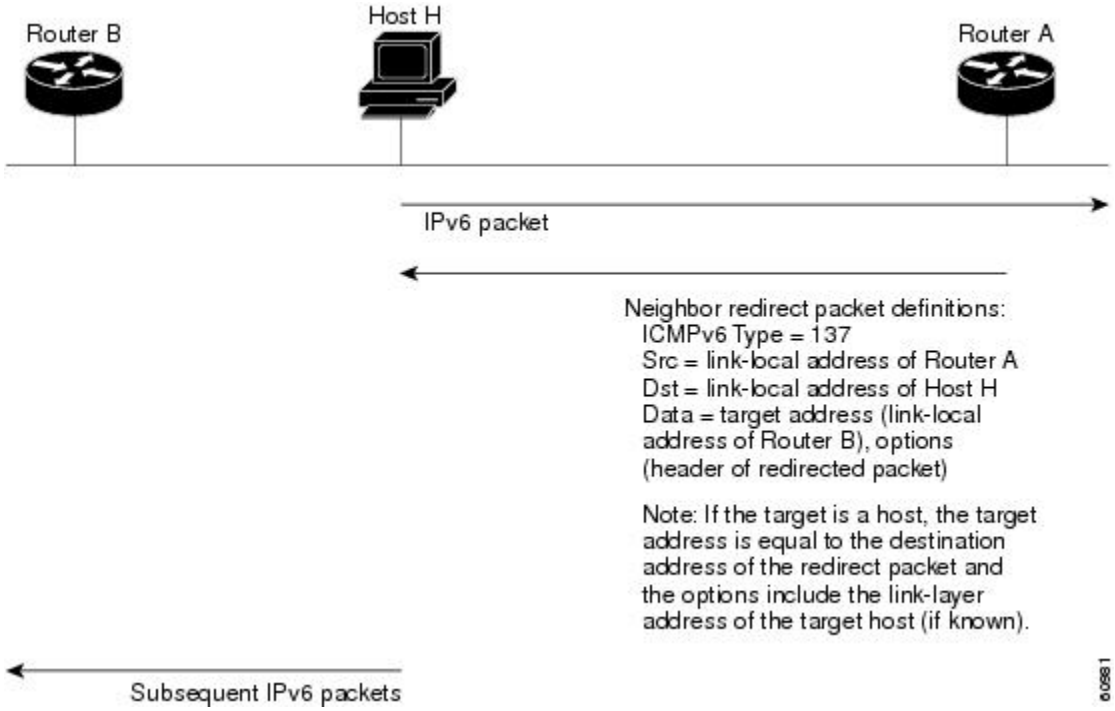
- Multiple devices that route to distinct sets of prefixes—Redirects (sent by nonoptimal devices for a destination) mean that hosts can choose any device and the system will work. However, traffic patterns may mean that choosing one of the devices would lead to considerably fewer redirects.
- Accidentally deploying a new device—Deploying a new device before it has been fully configured could lead to hosts adopting the new device as a default device and traffic disappearing. Network managers may want to indicate that some devices are more preferred than others.
- Multihomed situations—Multihomed situations may become more common, because of multiple physical links and because of the use of tunneling for IPv6 transport. Some of the devices may not provide full default routing because they route only to the 6-to-4 prefix or they route only to a corporate intranet. These situations cannot be resolved with redirects, which operate only over a single link.

The default router preference (DRP) feature provides a basic preference metric (low, medium, or high) for default devices. The DRP of a default device is signaled in unused bits in RA messages. This extension is backward compatible, both for devices (setting the DRP bits) and hosts (interpreting the DRP bits). These bits are ignored by hosts that do not implement the DRP extension. Similarly, the values sent by devices that do not implement the DRP extension will be interpreted by hosts that do implement it as indicating a “medium” preference. DRPs need to be configured manually.

IPv6 Neighbor Redirect Message

A value of 137 in the type field of the ICMP packet header identifies an IPv6 neighbor redirect message. Devices send neighbor redirect messages to inform hosts of better first-hop nodes on the path to a destination (see the figure below).

Figure 14 IPv6 Neighbor Discovery: Neighbor Redirect Message



Note A device must be able to determine the link-local address for each of its neighboring devices in order to ensure that the target address (the final destination) in a redirect message identifies the neighbor device by its link-local address. For static routing, the address of the next-hop device should be specified using the link-local address of the device; for dynamic routing, all IPv6 routing protocols must exchange the link-local addresses of neighboring devices.

After forwarding a packet, a device should send a redirect message to the source of the packet under the following circumstances:

- The destination address of the packet is not a multicast address.
- The packet was not addressed to the device.
- The packet is about to be sent out the interface on which it was received.
- The device determines that a better first-hop node for the packet resides on the same link as the source of the packet.
- The source address of the packet is a global IPv6 address of a neighbor on the same link, or a link-local address.

Use the **ipv6 icmp error-interval** command to limit the rate at which the device generates all IPv6 ICMP error messages, including neighbor redirect messages, which ultimately reduces link-layer congestion.

Note A device must not update its routing tables after receiving a neighbor redirect message, and hosts must not originate neighbor redirect messages.

Per-Interface Neighbor Discovery Cache Limit

The number of entries in the Neighbor Discovery cache can be limited by interface. Once the limit is reached, no new entries are allowed. The per-interface Neighbor Discovery cache limit function can be used to prevent any particular customer attached to an interface from overloading the Neighbor Discovery cache, whether intentionally or unintentionally.

When this feature is enabled globally, a common per-interface cache size limit is configured on all interfaces on the device. When this feature is enabled per interface, a cache size limit is configured on the associated interface. The per-interface limit overrides any globally configured limit.

Link, Subnet, and Site Addressing Changes

This section describes the IPv6 stateless autoconfiguration and general prefix features, which can be used to manage link, subnet, and site addressing changes.

IPv6 Stateless Autoconfiguration

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

Nodes can connect to a network and automatically generate global IPv6 addresses without the need for manual configuration or help of a server, such as a Dynamic Host Configuration Protocol (DHCP) server. With IPv6, a device on the link advertises any global prefixes in Router Advertisement (RA) messages, as well as its willingness to function as a default device for the link. RA messages are sent periodically and in response to device solicitation messages, which are sent by hosts at system startup.

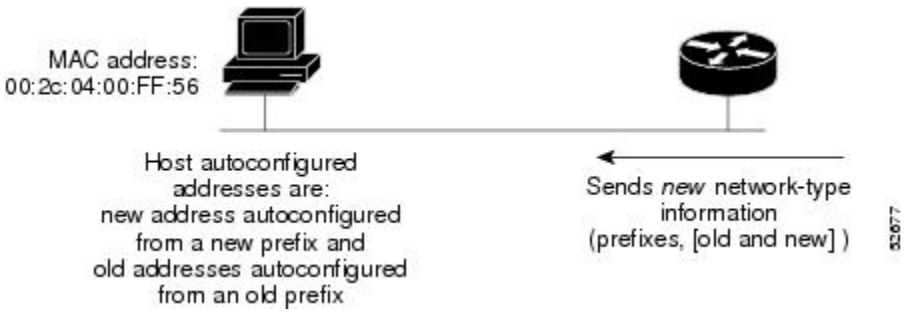
A node on the link can automatically configure global IPv6 addresses by appending its interface identifier (64 bits) to the prefixes (64 bits) included in the RA messages. The resulting 128-bit IPv6 addresses configured by the node are then subjected to duplicate address detection to ensure their uniqueness on the link. If the prefixes advertised in the RA messages are globally unique, then the IPv6 addresses configured by the node are also guaranteed to be globally unique. Device solicitation messages, which have a value of 133 in the Type field of the ICMP packet header, are sent by hosts at system startup so that the host can immediately autoconfigure without needing to wait for the next scheduled RA message.

Simplified Network Renumbering for IPv6 Hosts

The strict aggregation of the global routing table requires that networks be renumbered when the service provider for the network is changed. When the stateless autoconfiguration functionality in IPv6 is used to renumber a network, the prefix from a new service provider is added to RA messages that are sent on the link. (The RA messages contain both the prefix from the old service provider and the prefix from the new

service provider.) Nodes on the link automatically configure additional addresses by using the prefix from the new service provider. The nodes can then use the addresses created from the new prefix and the existing addresses created from the old prefix on the link. Configuration of the lifetime parameters associated with the old and new prefixes means that nodes on the link can make the transition to using only addresses created from the new prefix. During a transition period, the old prefix is removed from RA messages and only addresses that contain the new prefix are used on the link (the renumbering is complete) (see the figure below).

Figure 15 IPv6 Network Renumbering for Hosts Using Stateless Autoconfiguration



IPv6 General Prefixes

The upper 64 bits of an IPv6 address are composed from a global routing prefix plus a subnet ID, as defined in RFC 3513. A general prefix (for example, /48) holds a short prefix, based on which a number of longer, more-specific prefixes (for example, /64) can be defined. When the general prefix is changed, all of the more-specific prefixes based on it will change, too. This function greatly simplifies network renumbering and allows for automated prefix definition.

For example, a general prefix might be 48 bits long (“/48”) and the more specific prefixes generated from it might be 64 bits long (“/64”). In the following example, the leftmost 48 bits of all the specific prefixes will be the same, and they are the same as the general prefix itself. The next 16 bits are all different.

```
 General prefix: 2001:DB8:2222::/48
Specific prefix: 2001:DB8:2222:0000::/64
Specific prefix: 2001:DB8:2222:0001::/64
Specific prefix: 2001:DB8:2222:4321::/64
Specific prefix: 2001:DB8:2222:7744::/64
```

General prefixes can be defined in several ways:

- Manually
- Based on a 6to4 interface
- Dynamically, from a prefix received by a Dynamic Host Configuration Protocol (DHCP) for IPv6 prefix delegation client

More specific prefixes, based on a general prefix, can be used when configuring IPv6 on an interface.

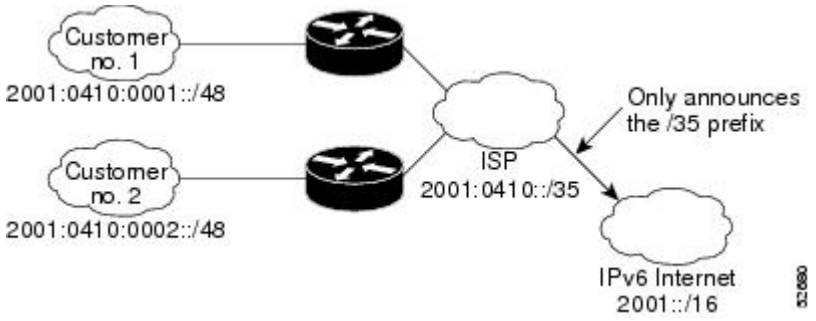
DHCP for IPv6 Prefix Delegation

DHCP for IPv6 can be used in environments to deliver stateful and stateless information. For further information about this feature, see *Implementing DHCP for IPv6*.

IPv6 Prefix Aggregation

The aggregatable nature of the IPv6 address space enables an IPv6 addressing hierarchy. For example, an enterprise can subdivide a single IPv6 prefix from a service provider into multiple, longer prefixes for use within its internal network. Conversely, a service provider can aggregate all of the prefixes of its customers into a single, shorter prefix that the service provider can then advertise over the IPv6 internet (see the figure below).

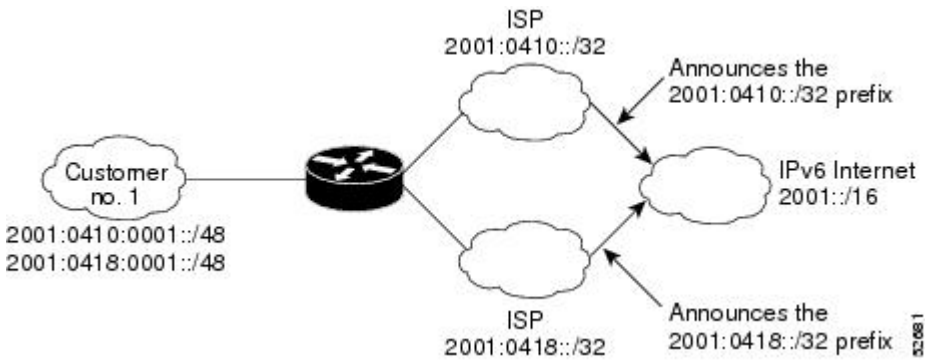
Figure 16 IPv6 Prefix Aggregation



IPv6 Site Multihoming

Multiple IPv6 prefixes can be assigned to networks and hosts. Having multiple prefixes assigned to a network allows that network to connect easily to multiple ISPs without breaking the global routing table (see the figure below).

Figure 17 IPv6 Site Multihoming



IPv6 Data Links

In IPv6 networks, a data link is a network sharing a particular link-local prefix. Data links are networks arbitrarily segmented by a network administrator in order to provide a multilevel, hierarchical routing structure while shielding the subnetwork from the addressing complexity of attached networks. The function of a subnetwork in IPv6 is similar to a subnetwork in IPv4. A subnetwork prefix is associated with one data link; multiple subnetwork prefixes may be assigned to the same data link.

The following data links are supported for IPv6: ATM permanent virtual circuit (PVC) and ATM LANE, dynamic packet transport (DPT), Ethernet, Fast Ethernet, FDDI, Frame Relay PVC, Gigabit Ethernet, Cisco High-Level Data Link Control (HDLC), ISDN, PPP over Packet over SONET (PoS), and serial interfaces.

IPv6 for Cisco Software Support for Wide-Area Networking Technologies

IPv6 for Cisco software supports wide-area networking technologies such as ATM PVCs, Frame Relay PVCs, Cisco HDLC, ISDN, PoS, and serial (synchronous and asynchronous) interface types. These technologies function the same in IPv6 as they do in IPv4.

IPv6 Addresses and PVCs

Broadcast and multicast are used in LANs to map protocol (network layer) addresses to the hardware addresses of remote nodes (hosts and devices). Because using broadcast and multicast to map network layer addresses to hardware addresses in circuit-based WANs such as ATM and Frame Relay networks is difficult to implement, these networks use implicit, explicit, and dynamic mappings for the network layer addresses of remote nodes and the PVCs used to reach the addresses.

Assigning an IPv6 address to an interface by using the **ipv6 address** command defines the IPv6 addresses for the interface and the network that is directly connected to the interface. If only one PVC is terminated on the interface (the interface is a point-to-point interface), there is an implicit mapping between all of the IPv6 addresses on the network and the PVC used to reach the addresses (no additional address mappings are needed). If several PVCs are terminated on the interface (the interface is a point-to-multipoint interface), the **protocol ipv6** command (for ATM networks) or the **frame-relay map ipv6** command (for Frame Relay networks) is used to configure explicit mappings between the IPv6 addresses of the remote nodes and the PVCs used to reach the addresses.

Note Given that IPv6 supports multiple address types, and depending on which applications or protocols are configured on a point-to-multipoint interface, you may need to configure multiple explicit mappings between the IPv6 addresses of the interface and the PVC used to reach the addresses. For example, explicitly mapping both the link-local and global IPv6 address of a point-to-multipoint interface to the PVC on which the interface terminates ensures that the Interior Gateway Protocol (IGP) configured on the interface forwards traffic to and from the PVC correctly.

Routed Bridge Encapsulation for IPv6

Routed bridge encapsulation (RBE) provides a mechanism for routing a protocol from a bridged interface to another routed or bridged interface. RBE for IPv6 can be used on ATM point-to-point subinterfaces that are configured for IPv6 half-bridging. Routing of IP packets and IPv6 half-bridging, bridging, PPP over Ethernet (PPPoE), or other Ethernet 802.3-encapsulated protocols can be configured on the same subinterface.

IPv6 Redirect Messages

The IPv6 Redirect Messages feature enables a device to send ICMP IPv6 neighbor redirect messages to inform hosts of better first hop nodes (devices or hosts) on the path to a destination.

IPv6 on BVI Interfaces for Bridging and Routing

Integrated routing and bridging (IRB) enables users to route a given protocol between routed interfaces and bridge groups or route a given protocol between bridge groups. Specifically, local or unroutable traffic will be bridged among the bridged interfaces in the same bridge group, while routable traffic will be routed to other routed interfaces or bridge groups. If you want both bridging and routing capabilities, IRB is required. If you want only bridging, you must disable routing. To disable the routing function for IPv6, you must configure the **no ipv6 unicast-routing** command.

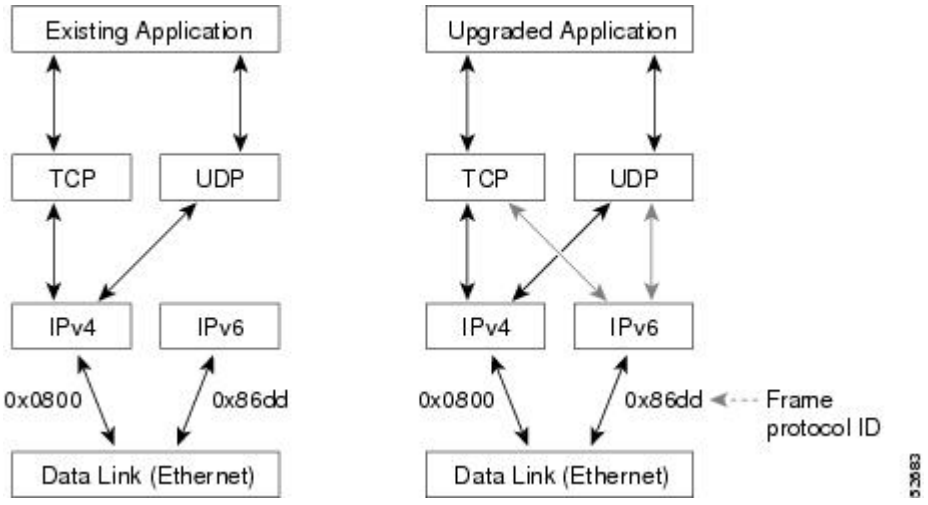
IPv6 is supported in the bridge virtual interface (BVI), which is the IPv4 interface for bridged interfaces. Because bridging is in the data link layer and routing is in the network layer, they have different protocol configuration models to follow. In the basic IPv4 model, for example, all bridged interfaces should belong to the same network, while each routed interface represents a distinct network. Routed traffic is destined for the device, while bridged traffic is never destined for the device. Using BVI avoids the confusion of which protocol configuration model to use when both bridging and routing a given protocol in the same bridge group.

Note BVIs in IPv6 are not supported with Network Address Translation--Protocol Translation (NAT-PT) and wireless interfaces Dot11Radio.

Dual IPv4 and IPv6 Protocol Stacks

The dual IPv4 and IPv6 protocol stack technique can be used to transition to IPv6. It enables gradual, one-by-one upgrades to applications running on nodes. Applications running on nodes are upgraded to make use of the IPv6 protocol stack. Applications that are not upgraded (for example, they support only the IPv4 protocol stack) can coexist with upgraded applications on a node. New and upgraded applications make use of both the IPv4 and IPv6 protocol stacks (see the figure below).

Figure 18 Dual IPv4 and IPv6 Protocol Stack Technique

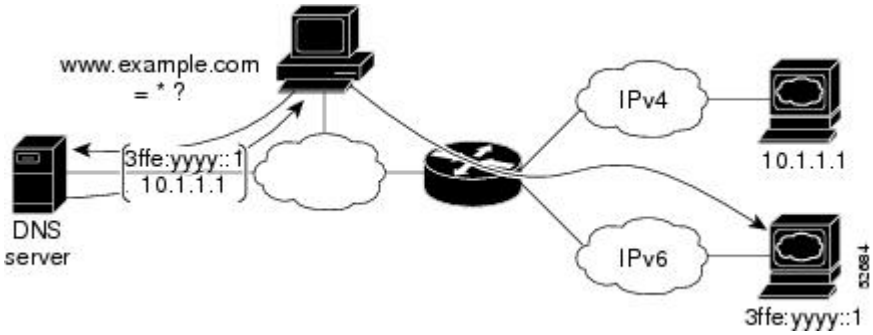


One application program interface (API) supports both IPv4 and IPv6 addresses and DNS requests. An application can be upgraded to the new API and still use only the IPv4 protocol stack. The Cisco software

supports the dual IPv4 and IPv6 protocol stack technique. When an interface is configured with both an IPv4 and an IPv6 address, the interface will forward both IPv4 and IPv6 traffic.

In the figure below, an application that supports dual IPv4 and IPv6 protocol stacks requests all available addresses for the destination hostname www.example.com from a DNS server. The DNS server replies with all available addresses (both IPv4 and IPv6 addresses) for www.example.com. The application chooses an address (in most cases, IPv6 addresses are the default choice), and connects the source node to the destination using the IPv6 protocol stack.

Figure 19 Dual IPv4 and IPv6 Protocol Stack Applications



How to Implement IPv6 Addressing and Basic Connectivity

Configuring IPv6 Addressing and Enabling IPv6 Routing

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. Do one of the following:
   - **ipv6 address** *ipv6-prefix/prefix-length* **eui-64**
   - **ipv6 address** *ipv6-prefix/prefix-length* **link-local**
   - **ipv6 address** *ipv6-prefix/prefix-length* **anycast**
   - **ipv6 enable**
5. **exit**
6. **ipv6 unicast-routing**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface ethernet 0/0`	Specifies an interface type and number, and places the device in interface configuration mode.

	Command or Action	Purpose

Step 4	Do one of the following:
	• **ipv6 address** *ipv6-prefix/prefix-length* **eui-64**
	• **ipv6 address** *ipv6-prefix/prefix-length* **link-local**
	• **ipv6 address** *ipv6-prefix/prefix-length* **anycast**
	• **ipv6 enable**
	Example:
	`Device(config-if)# ipv6 address 2001:DB8:0:1::/64 eui-64`
	Example:
	`Device(config-if)# ipv6 address FE80::260:3EFF:FE11:6770 link-local`
	Example:
	`Device(config-if) ipv6 address 2001:DB8:1:1:FFFF:FFFF:FFFF:FFFE/64 anycast`
	Example:
	`Device(config-if)# ipv6 enable`
		Specifies an IPv6 network assigned to the interface and enables IPv6 processing on the interface.
		or
		Specifies an IPv6 address assigned to the interface and enables IPv6 processing on the interface.
		or
		Automatically configures an IPv6 link-local address on the interface while also enabling the interface for IPv6 processing.
		or
		Enable IPv6 processing on an interface that has not been configured with an explicit IPv6 address.
		• Specifying the **ipv6 address eui-64** command configures global IPv6 addresses with an interface identifier (ID) in the low-order 64 bits of the IPv6 address. Only the 64-bit network prefix for the address needs to be specified; the last 64 bits are automatically computed from the interface ID.
		• Specifying the **ipv6 address link-local** command configures a link-local address on the interface that is used instead of the link-local address that is automatically configured when IPv6 is enabled on the interface.
		• Specifying the **ipv6 address anycast** command adds an IPv6 anycast address.

Step 5	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode, and returns the device to global configuration mode.
Step 6	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.

- Configuring a Neighbor Discovery Cache Limit, page 69
- Customizing the Parameters for IPv6 Neighbor Discovery, page 71

Configuring a Neighbor Discovery Cache Limit

Configuring a Neighbor Discovery Cache Limit on a Specified Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 nd cache interface-limit** *size* [**log** *rate*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/0/0`	Specifies an interface type and number, and places the device in interface configuration mode.
Step 4	**ipv6 nd cache interface-limit** *size* [**log** *rate*] **Example:** `Device(config-if)# ipv6 nd cache interface-limit 1`	Configures a Neighbor Discovery cache limit on a specified interface on the device. • Issuing this command overrides any configuration that may have been created by issuing the **ipv6 nd cache interface-limit** in global configuration mode.

Configuring a Neighbor Discovery Cache Limit on All Device Interfaces

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 nd cache interface-limit** *size* [**log** *rate*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 nd cache interface-limit** *size* [**log** *rate*] **Example:** `Device(config)# ipv6 nd cache interface-limit 4`	Configures a neighbor discovery cache limit on all interfaces on the device.

Customizing the Parameters for IPv6 Neighbor Discovery

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 nd nud retry** *base interval max-attempts*
5. **ipv6 nd cache expire** *expire-time-in-seconds* [**refresh**]
6. **ipv6 nd na glean**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface Ethernet 1/0`	Specifies an interface type and number, and places the device in interface configuration mode.
Step 4	**ipv6 nd nud retry** *base interval max-attempts* **Example:** `Device(config-if)# ipv6 nd nud retry 1 1000 3`	Configures the number of times neighbor unreachability detection (NUD) resends neighbor solicitations.
Step 5	**ipv6 nd cache expire** *expire-time-in-seconds* [**refresh**] **Example:** `Device(config-if)# ipv6 nd cache expire 7200`	Configures the length of time before an IPv6 neighbor discovery cache entry expires.
Step 6	**ipv6 nd na glean** **Example:** `Device(config-if)# ipv6 nd na glean`	Configures ND to glean an entry from an unsolicited neighbor advertisement (NA).

Defining and Using IPv6 General Prefixes

General prefixes can be defined in several ways:

- Manually
- Based on a 6to4 interface
- Dynamically, from a prefix received by a DHCP for IPv6 prefix delegation client

More specific prefixes, based on a general prefix, can be used when configuring IPv6 on an interface.

- Defining a General Prefix Manually, page 73
- Defining a General Prefix Based on a 6to4 Interface, page 73
- Defining a General Prefix with the DHCP for IPv6 Prefix Delegation Client Function, page 74
- Using a General Prefix in IPv6, page 74

Defining a General Prefix Manually

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 general-prefix** *prefix-name* {*ipv6-prefix/prefix-length* | **6to4** *interface-type interface-number*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 general-prefix** *prefix-name* {*ipv6-prefix/prefix-length*	**6to4** *interface-type interface-number*} **Example:** `Device(config)# ipv6 general-prefix my-prefix 2001:DB8:2222::/48`	Defines a general prefix for an IPv6 address.

Defining a General Prefix Based on a 6to4 Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 general-prefix** *prefix-name* {*ipv6-prefix/prefix-length* | **6to4** *interface-type interface-number*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 general-prefix** *prefix-name* {*ipv6-prefix/prefix-length* | **6to4** *interface-type interface-number*} **Example:** `Device(config)# ipv6 general-prefix my-prefix 6to4 ethernet 0`	Defines a general prefix for a 6to4 address.

Defining a General Prefix with the DHCP for IPv6 Prefix Delegation Client Function

You can define a general prefix dynamically using the DHCP for IPv6 prefix delegation client function. For information on how to perform this task, see the Implementing DHCP for IPv6 module.

Using a General Prefix in IPv6

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 address** { *ipv6-address/prefix-length* | *prefix-name sub-bits/prefix-length*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface ethernet 0/0`	Specifies an interface type and number, and places the device in interface configuration mode.
Step 4	**ipv6 address** { *ipv6-address/prefix-length* | *prefix-name sub-bits/ prefix-length*} **Example:** `Device(config-if) ipv6 address my-prefix 2001:DB8:0:7272::/64`	Configures an IPv6 prefix name for an IPv6 address and enables IPv6 processing on the interface.

Configuring an Interface to Support the IPv4 and IPv6 Protocol Stacks

When an interface in a Cisco networking device is configured with both an IPv4 and an IPv6 address, the interface forwards both IPv4 and IPv6 traffic; that is, the interface can send and receive data on both IPv4 and IPv6 networks.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **interface** *type number*
5. **ip address** *ip-address mask* [**secondary** [**vrf** *vrf-name*]]
6. **ipv6 address** {*ipv6-address /prefix-length* | *prefix-name sub-bits* / *prefix-length*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.
Step 4	**interface** *type number* **Example:** `Device(config)# interface ethernet 0`	Specifies the interface type and number, and enters interface configuration mode.
Step 5	**ip address** *ip-address mask* [**secondary** [**vrf** *vrf-name*]] **Example:** `Device(config-if)# ip address 192.168.99.1 255.255.255.0`	Specifies a primary or secondary IPv4 address for an interface.
Step 6	**ipv6 address** {*ipv6-address /prefix-length* | *prefix-name sub-bits / prefix-length*} **Example:** `Device(config-if)# ipv6 address 2001:DB8:c18:1::3/64`	Specifies the IPv6 network assigned to the interface and enables IPv6 processing on the interface.

Customizing IPv6 ICMP Rate Limiting

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 icmp error-interval** *milliseconds* [*bucketsize*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 icmp error-interval** *milliseconds* [*bucketsize*] **Example:** `Device(config)# ipv6 icmp error-interval 50 20`	Customizes the interval and bucket size for IPv6 ICMP error messages.

Configuring the DRP Extension for Traffic Engineering

Perform this task to configure the DRP extension to RAs, which signals the preference value of a default device.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 nd router-preference** {**high** | **medium** | **low**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface ethernet 0`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**ipv6 nd router-preference** {**high** | **medium** | **low**} **Example:** `Device(config-if)# ipv6 nd router-preference high`	Configures a DRP for a device on a specific interface.

Configuring Cisco Express Forwarding and Distributed Cisco Express Forwarding Switching for IPv6

Configuring Cisco Express Forwarding Switching on Distributed and Nondistributed Architecture Platforms

Cisco Express Forwarding is designed for nondistributed architecture platforms. Distributed Cisco Express Forwarding is designed for distributed architecture platforms. Nondistributed platforms do not support distributed Cisco Express Forwarding; however, some distributed platforms support both Cisco Express Forwarding and distributed Cisco Express Forwarding.

To enable the device to forward Cisco Express Forwarding and distributed Cisco Express Forwarding traffic, use the **ipv6 unicast-routing** command to configure the forwarding of IPv6 unicast datagrams globally on the device, and use the **ipv6 address** command to configure IPv6 address and IPv6 processing on an interface.

You must enable Cisco Express Forwarding for IPv4 globally on the device by using the **ip cef** command before enabling Cisco Express Forwarding for IPv6 globally on the device.

You must enable distributed Cisco Express Forwarding for IPv4 by using the **ip cef distributed** command before enabling distributed Cisco Express Forwarding for IPv6.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. Do one of the following:
   - **ipv6 cef**
   - **ipv6 cef distributed**
4. **ipv6 cef accounting** [**non-recursive** | **per-prefix** | **prefix-length**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	Do one of the following:
	• **ipv6 cef**
	• **ipv6 cef distributed**
	
	Example:
	`Device(config)# ipv6 cef`
	
	Example:
	`Device(config)# ipv6 cef distributed`
		Enables Cisco Express Forwarding globally on the device.
		or
		Enables distributed Cisco Express Forwarding globally on the device.

Command or Action	Purpose
Step 4 **ipv6 cef accounting** [**non-recursive** | **per-prefix** | **prefix-length**] **Example:** `Device(config)# ipv6 cef accounting`	Enables Cisco Express Forwarding and distributed Cisco Express Forwarding network accounting globally on the device. • Network accounting for Cisco Express Forwarding and distributed Cisco Express Forwarding enables you to better understand Cisco Express Forwarding traffic patterns within your network by collecting statistics specific to Cisco Express Forwarding and distributed Cisco Express Forwarding traffic. For example, network accounting for Cisco Express Forwarding and distributed Cisco Express Forwarding enables you to collect information such as the number of packets and bytes switched to a destination or the number of packets switched through a destination. • The optional **per-prefix** keyword enables the collection of the number of packets and bytes express forwarded to an IPv6 destination or IPv6 prefix. • The optional **prefix-length** keyword enables the collection of the number of packets and bytes express forwarded to an IPv6 prefix length. **Note** When Cisco Express Forwarding is enabled globally on the device, accounting information is collected at the RP; when distributed Cisco Express Forwarding is enabled globally on the device, accounting information is collected at the line cards.

Enabling Unicast RPF

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 verify unicast source reachable-via** {**rx** | **any**} [**allow-default**] [**allow-self-ping**] [*access-list-name*]

DETAILED STEPS

Command or Action	Purpose
Step 1 **enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface atm 0`	Specifies an interface type and number, and places the device in interface configuration mode.
Step 4	**ipv6 verify unicast source reachable-via** {**rx** | **any**} [**allow-default**] [**allow-self-ping**] [*access-list-name*] **Example:** `Device(config-if)# ipv6 verify unicast source reachable-via any`	Verifies that a source address exists in the FIB table and enables uRPF.

Mapping Hostnames to IPv6 Addresses

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 host** *name* [*port*] *ipv6-address1* [*ipv6-address2...ipv6-address4*]
4. Do one of the following:
   - **ip domain name** [**vrf** *vrf-name*] *name*
   - **ip domain list** [**vrf** *vrf-name*] *name*
5. **ip name-server** [**vrf** *vrf-name*] *server-address1* [*server-address2...server-address6*]
6. **ip domain-lookup**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

Command or Action	Purpose
Step 2 **configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3 **ipv6 host** *name* [*port*] *ipv6-address1* [*ipv6-address2...ipv6-address4*] **Example:** `Device(config)# ipv6 host cisco-sj 2001:DB8:20:1::12`	Defines a static hostname-to-address mapping in the hostname cache. • You may find it easier to refer to network devices by symbolic names rather than numerical addresses (services such as Telnet can use hostnames or addresses). Hostnames and IPv6 addresses can be associated with one another through static or dynamic means. • Manually assigning hostnames to addresses is useful when dynamic mapping is not available.

Step 4 Do one of the following:
• **ip domain name** [**vrf** *vrf-name*] *name*
• **ip domain list** [**vrf** *vrf-name*] *name*
 	
Example:
`Device(config)# ip domain-name cisco.com`
 	
Example:
`Device(config)# ip domain list cisco1.com`
	(Optional) Defines a default domain name that the Cisco software will use to complete unqualified hostnames.
	or
	(Optional) Defines a list of default domain names to complete unqualified hostnames.
	• You can specify a default domain name that the Cisco software will use to complete domain name requests. You can specify either a single domain name or a list of domain names. Any hostname that does not contain a complete domain name will have the default domain name you specify appended to it before the name is looked up.
	Note The **ip domain name** and **ip domain list** commands are used to specify default domain names that can be used by both IPv4 and IPv6.

Step 5 **ip name-server** [**vrf** *vrf-name*] *server-address1* [*server-address2...server-address6*] **Example:** `Device(config)# ip name-server 2001:DB8::250:8bff:fee8:f800 2001:DB8:0:f004::1`	Specifies one or more hosts that supply name information. • Specifies one or more hosts (up to six) that can function as a name server to supply name information for DNS. **Note** The *server-address* argument can be either an IPv4 or IPv6 address.
Step 6 **ip domain-lookup** **Example:** `Device(config)# ip domain-lookup`	Enables DNS-based address translation. • DNS is enabled by default.

Mapping IPv6 Addresses to IPv6 ATM and Frame Relay Interfaces

Perform this task to map IPv6 addresses to ATM and Frame Relay PVCs. Specifically, the steps in this section explain how to explicitly map IPv6 addresses to the ATM and Frame Relay PVCs used to reach the addresses.

Note This task shows how to configure both ATM and Frame Relay PVCs. Many of the steps are labeled optional because many networks will require only one type of PVC to be configured. The steps in this section are not applicable to ATM LANE.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **pvc** [*name*] *vpi/vci* [**ces** | **ilmi** | **qsaal** | **smds** | **l2transport**]
5. **protocol ipv6** *ipv6-address* [[**no**] **broadcast**]
6. **exit**
7. **ipv6 address** *ipv6-address/prefix-length* **link-local**
8. **exit**
9. **interface** *type number*
10. **frame-relay map ipv6** *ipv6-address dlci* [**broadcast**] [**cisco**] [**ietf**] [**payload-compression** {**packet-by-packet** | **frf9 stac** [*hardware-options*] | **data-stream stac** [*hardware-options*]}]
11. **ipv6 address** *ipv6-address/prefix-length* **link-local**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Router(config)# interface atm 0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**pvc** [*name*] *vpi/vci* [**ces** | **ilmi** | **qsaal** | **smds** | **l2transport**] **Example:** `Router(config-if)# pvc 1/32`	(Optional) Creates or assigns a name to an ATM PVC and places the router in ATM VC configuration mode.
Step 5	**protocol ipv6** *ipv6-address* [[**no**] **broadcast**] **Example:** `Router(config-if-atm-vc)# protocol ipv6 2001:DB8:2222:1003::45`	(Optional) Maps the IPv6 address of a remote node to the PVC used to reach the address. • The *ipv6-address* argument must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons. • The optional [**no**] **broadcast** keywords indicate whether the map entry should be used when IPv6 multicast packets (not broadcast packets) are sent to the interface. Pseudobroadcasting is supported. The [**no**] **broadcast** keywords in the **protocol ipv6** command take precedence over the **broadcast** command configured on the same ATM PVC.
Step 6	**exit** **Example:** `Router(config-if-atm-vc)# exit`	Exits ATM VC configuration mode, and returns the router to interface configuration mode.
Step 7	**ipv6 address** *ipv6-address/prefix-length* **link-local** **Example:** `Router(config-if)# ipv6 address 2001:DB8:2222:1003::72/64 link-local`	Specifies an IPv6 network assigned to the interface and enables IPv6 processing on the interface. • In the context of this task, a link-local address of the node at the other end of the link is required for the IGP used in the network. • Specifying the **ipv6 address link-local**command configures a link-local address on the interface that is used instead of the link-local address that is automatically configured when IPv6 is enabled on the interface.
Step 8	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode, and returns the router to global configuration mode.

	Command or Action	Purpose
Step 9	**interface** *type number* **Example:** `Router(config)# interface serial 3`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 10	**frame-relay map ipv6** *ipv6-address dlci* [**broadcast**] [**cisco**] [**ietf**] [**payload-compression** {**packet-by-packet** | **frf9 stac** [*hardware-options*] | **data-stream stac** [*hardware-options*]}] **Example:** `Router(config-if)# frame-relay map ipv6 FE80::E0:F727:E400:A 17 broadcast`	(Optional) Maps the IPv6 address of a remote node to the data-link connection identifier (DLCI) of the PVC used to reach the address.
Step 11	**ipv6 address** *ipv6-address/prefix-length* **link-local** **Example:** `Router(config-if)# ipv6 address 2001:DB8:2222:1044::46/64 link-local`	Specifies an IPv6 network assigned to the interface and enables IPv6 processing on the interface. • In the context of this task, a link-local address of the node at the other end of the link is required for the IGP used in the network. • Specifying the **ipv6 address link-local** command configures a link-local address on the interface that is used instead of the link-local address that is automatically configured when IPv6 is enabled on the interface.

Displaying IPv6 Redirect Messages

SUMMARY STEPS

1. **enable**
2. **show ipv6 interface** [**brief**] [*type number*] [**prefix**]
3. **show ipv6 neighbors** [*interface-type interface-number* | *ipv6-address* | *ipv6-hostname* | **statistics**]
4. **show ipv6 route** [*ipv6-address* | *ipv6-prefix* / *prefix-length* | *protocol* | *interface-type interface-number*]
5. **show ipv6 traffic**
6. **show atm map**
7. **show hosts** [**vrf** *vrf-name* | **all** | *hostname* | **summary**]
8. **show running-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ipv6 interface** [**brief**] [*type number*] [**prefix**] **Example:** `Device# show ipv6 interface ethernet 0`	Displays the usability status of interfaces configured for IPv6.
Step 3	**show ipv6 neighbors** [*interface-type interface-number* | *ipv6-address* | *ipv6-hostname* | **statistics**] **Example:** `Device# show ipv6 neighbors ethernet 2`	Displays IPv6 neighbor discovery cache information.
Step 4	**show ipv6 route** [*ipv6-address* | *ipv6-prefix* / *prefix-length* | *protocol* | *interface-type interface-number*] **Example:** `Device# show ipv6 route`	Displays the current contents of the IPv6 routing table.
Step 5	**show ipv6 traffic** **Example:** `Device# show ipv6 traffic`	Displays statistics about IPv6 traffic.
Step 6	**show atm map** **Example:** `Device# show atm map`	Displays the list of all configured ATM static maps to remote hosts on an ATM network and on ATM bundle maps.
Step 7	**show hosts** [**vrf** *vrf-name* | **all** | *hostname* | **summary**] **Example:** `Device# show hosts`	Displays the default domain name, the style of name lookup service, a list of name server hosts, and the cached list of hostnames and addresses.

Command or Action	Purpose
Step 8 **show running-config** **Example:** `Device# show running-config`	Displays the current configuration running on the device.

- Examples, page 87

Examples

Sample Output from the show ipv6 interface Command

In the following example, the **show ipv6 interface** command is used to verify that IPv6 addresses are configured correctly for Ethernet interface 0. Information is also displayed about the status of IPv6 neighbor redirect messages, IPv6 neighbor discovery messages, and stateless autoconfiguration.

```
Router# show ipv6 interface ethernet 0

Ethernet0 is up, line protocol is up
  IPv6 is stalled, link-local address is FE80::1
  Global unicast address(es):
    2001:DB8:2000::1, subnet is 2001:DB8:2000::/64
    2001:DB8:3000::1, subnet is 2001:DB8:3000::/64
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FF00:1
  MTU is 1500 bytes
  ICMP error messages limited to one every 100 milliseconds
  ICMP redirects are enabled
  ND DAD is enabled, number of DAD attempts: 1
  ND reachable time is 30000 milliseconds
  ND advertised reachable time is 0 milliseconds
  ND advertised retransmit interval is 0 milliseconds
  ND router advertisements are sent every 200 seconds
  ND router advertisements live for 1800 seconds
  Hosts use stateless autoconfig for addresses.
```

Sample Output from the show ipv6 neighbors Command

In the following example, the **show ipv6 neighbors** command is used to display IPv6 neighbor discovery cache information. A hyphen (-) in the Age field of the command output indicates a static entry. The following example displays IPv6 neighbor discovery cache information for Ethernet interface 2:

```
Router# show ipv6 neighbors ethernet 2

IPv6 Address                              Age Link-layer Addr State Interface
2001:DB8:0:4::2                            0 0003.a0d6.141e  REACH Ethernet2
FE80::XXXX:A0FF:FED6:141E                  0 0003.a0d6.141e  REACH Ethernet2
2001:DB8:1::45a                            - 0002.7d1a.9472  REACH Ethernet2
```

Sample Output from the show ipv6 route Command

When the *ipv6-address* or *ipv6-prefix*/*prefix-length* argument is specified, only route information for that address or network is displayed. The following is sample output from the **show ipv6 route** command when entered with the IPv6 prefix 2001:DB8::/35:

```
Router# show ipv6 route 2001:DB8::/35

IPv6 Routing Table - 261 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
B 2001:DB8::/35 [20/3]
  via FE80::60:5C59:9E00:16, Tunnel1
```

Sample Output from the show ipv6 traffic Command

In the following example, the **show ipv6 traffic** command is used to display ICMP rate-limited counters:

```
Router# show ipv6 traffic

ICMP statistics:
  Rcvd: 188 input, 0 checksum errors, 0 too short
        0 unknown info type, 0 unknown error type
        unreach: 0 routing, 0 admin, 0 neighbor, 0 address, 0 port
        parameter: 0 error, 0 header, 0 option
        0 hopcount expired, 0 reassembly timeout,0 too big
        0 echo request, 0 echo reply
        0 group query, 0 group report, 0 group reduce
        1 router solicit, 175 router advert, 0 redirects
        0 neighbor solicit, 12 neighbor advert
  Sent: 7376 output, 56 rate-limited
        unreach: 0 routing, 15 admin, 0 neighbor, 0 address, 0 port
        parameter: 0 error, 0 header, 0 option
        0 hopcount expired, 0 reassembly timeout,0 too big
        15 echo request, 0 echo reply
        0 group query, 0 group report, 0 group reduce
        0 router solicit, 7326 router advert, 0 redirects
        2 neighbor solicit, 22 neighbor advert
```

Sample Output from the show frame-relay map Command

In the following example, the **show frame-relay map** command is used to verify that the IPv6 address of a remote node is mapped to the DLCI of the PVC used to reach the address. The following example shows that the link-local and global IPv6 addresses (FE80::E0:F727:E400:A and 2001:DB8:2222:1044::73; FE80::60:3E47:AC8:8 and 2001.DB8:2222:1044::72) of two remote nodes are explicitly mapped to DLCI 17 and DLCI 19, respectively. Both DLCI 17 and DLCI 19 are terminated on interface serial 3 of this node; therefore, interface serial 3 of this node is a point-to-multipoint interface.

```
Router# show frame-relay map

Serial3 (up): ipv6 FE80::E0:F727:E400:A dlci 17(0x11,0x410), static,
              broadcast, CISCO, status defined, active
Serial3 (up): ipv6 2001:DB8:2222:1044::72 dlci 19(0x13,0x430), static,
              CISCO, status defined, active
Serial3 (up): ipv6 2001:DB8:2222:1044::73 dlci 17(0x11,0x410), static,
              CISCO, status defined, active
Serial3 (up): ipv6 FE80::60:3E47:AC8:8 dlci 19(0x13,0x430), static,
              broadcast, CISCO, status defined, active
```

Sample Output from the show atm map Command

In the following example, the **show atm map** command is used to verify that the IPv6 address of a remote node is mapped to the PVC used to reach the address. The following example shows that the link-local and

global IPv6 addresses (FE80::60:3E47:AC8:C and 2001:DB8:2222:1003::72, respectively) of a remote node are explicitly mapped to PVC 1/32 of ATM interface 0:

```
Router# show atm map

Map list ATM0pvc1 : PERMANENT
ipv6 FE80::60:3E47:AC8:C maps to VC 1, VPI 1, VCI 32, ATM0
        , broadcast
ipv6 2001:DB8:2222:1003::72 maps to VC 1, VPI 1, VCI 32, ATM0
```

Sample Output from the show hosts Command

The state of the name lookup system on the DHCP for IPv6 client can be displayed with the **show hosts** command:

```
Router# show hosts

Default domain is not set
Domain list:example.com
Name/address lookup uses domain service
Name servers are 2001:DB8:A:B::1, 2001:DB8:3000:3000::42
Codes:UN - unknown, EX - expired, OK - OK, ?? - revalidate
       temp - temporary, perm - permanent
       NA - Not Applicable None - Not defined
Host                     Port  Flags      Age Type   Address(es)
sdfasfd                  None  (temp, UN)  0  IPv6
```

Sample Output from the show running-config Command

In the following example, the **show running-config** command is used to verify that IPv6 processing of packets is enabled globally on the router and on applicable interfaces, and that an IPv6 address is configured on applicable interfaces:

```
Router# show running-config

Building configuration...
Current configuration : 22324 bytes
!
! Last configuration change at 14:59:38 PST Tue Jan 16 2001
! NVRAM config last updated at 04:25:39 PST Tue Jan 16 2001 by bird
!
hostname Router
!
ipv6 unicast-routing
!
interface Ethernet0
 no ip route-cache
 no ip mroute-cache
 no keepalive
 media-type 10BaseT
    ipv6 address 2001:DB8:0:1::/64 eui-64
!
```

In the following example, the **show running-config** command is used to verify that Cisco Express Forwarding and network accounting for Cisco Express Forwarding have been enabled globally on a nondistributed architecture platform, and that Cisco Express Forwarding has been enabled on an IPv6 interface. The following output shows that both that Cisco Express Forwarding and network accounting for Cisco Express Forwarding have been enabled globally on the router, and that Cisco Express Forwarding has also been enabled on Ethernet interface 0:

```
Router# show running-config

Building configuration...
Current configuration : 22324 bytes
!
```

```
! Last configuration change at 14:59:38 PST Tue Jan 16 2001
! NVRAM config last updated at 04:25:39 PST Tue Jan 16 2001 by bird
!
hostname Router
!
ip cef
ipv6 unicast-routing
ipv6 cef
ipv6 cef accounting prefix-length
!
!
interface Ethernet0
 ip address 10.4.9.11 255.0.0.0
 media-type 10BaseT
 ipv6 address 2001:DB8:C18:1::/64 eui-64
!
```

In the following example, the **show running-config** command is used to verify that distributed Cisco Express Forwarding and network accounting for distributed Cisco Express Forwarding have been enabled globally on a distributed architecture platform, such as the Cisco 7500 series routers. The following example shows that both distributed Cisco Express Forwarding and network accounting for Cisco Express Forwarding have been enabled globally on the router.

Note Distributed Cisco Express Forwarding is enabled by default on the GSRs and disabled by default on the Cisco 7500 series routers. Therefore, output from the **show running-config** command on the GSRs does not show whether distributed Cisco Express Forwarding is configured globally on the router. The following output is from a Cisco 7500 series router.

```
Router# show running-config

Building configuration...
Current configuration : 22324 bytes
!
! Last configuration change at 14:59:38 PST Tue Jan 16 2001
! NVRAM config last updated at 04:25:39 PST Tue Jan 16 2001 by bird
!
hostname Router
!
ip cef distributed
ipv6 unicast-routing
ipv6 cef distributed
ipv6 cef accounting prefix-length
```

In the following example, the **show running-config** command is used to verify static hostname-to-address mappings, default domain names, and name servers in the hostname cache, and to verify that the DNS service is enabled:

```
Router# show running-config

Building configuration...
!
ipv6 host cisco-sj 2001:DB8:20:1::12
!
ip domain-name cisco.com
ip domain-lookup
ip name-server 2001:DB8:C01F:768::1
```

Configuration Examples for Implementing IPv6 Addressing and Basic Connectivity

Example: IPv6 Addressing and IPv6 Routing Configuration

In this example, IPv6 is enabled on the device with both a link-local address and a global address based on the IPv6 prefix 2001:DB8:c18:1::/64. The EUI-64 interface ID is used in the low-order 64 bits of both addresses. Output from the **show ipv6 interface** command is included to show how the interface ID (260:3EFF:FE47:1530) is appended to the link-local prefix FE80::/64 of Ethernet interface 0.

```
ipv6 unicast-routing
interface ethernet 0
  ipv6 address 2001:DB8:c18:1::/64 eui-64

Device# show ipv6 interface ethernet 0

Ethernet0 is up, line protocol is up
 IPv6 is enabled, link-local address is FE80::260:3EFF:FE47:1530
  Global unicast address(es):
    2001:DB8:C18:1:260:3EFF:FE47:1530, subnet is 2001:DB8:C18:1::/64
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FF47:1530
    FF02::9
  MTU is 1500 bytes
  ICMP error messages limited to one every 500 milliseconds
  ND reachable time is 30000 milliseconds
  ND advertised reachable time is 0 milliseconds
  ND advertised retransmit interval is 0 milliseconds
  ND router advertisements are sent every 200 seconds
  ND router advertisements live for 1800 seconds
  Hosts use stateless autoconfig for addresses.
```

In the following example, multiple IPv6 global addresses within the prefix 2001:DB8::/64 are configured on Ethernet interface 0:

```
interface ethernet 0
 ipv6 address 2001:DB8::1/64
 ipv6 address 2001:DB8::/64 eui-64
```


Example: Customizing the Parameters for IPv6 Neighbor Discovery

In the following example, IPv6 ND NA gleaning is enabled and the IPv6 ND cache expiry is set to 7200 seconds (2 hours):

```
interface Port-channel189
```

```
 no ip address
 ipv6 address FC07::789:1:0:0:3/64
 ipv6 nd reachable-time 2700000
 ipv6 nd na glean
 ipv6 nd cache expire 7200
 no ipv6 redirects
 standby version 2
 standby 2 ipv6 FC07::789:1:0:0:1/64
 standby 2 priority 150
 standby 2 preempt
```

Example: Dual-Protocol Stack Configuration

This example shows how to enable the forwarding of IPv6 unicast datagrams globally on the device and configures Ethernet interface 0 with both an IPv4 address and an IPv6 address:

```
ipv6 unicast-routing
interface Ethernet 0
  ip address 192.168.99.1 255.255.255.0
  ipv6 address 2001:DB8:c18:1::3/64
```

Example: IPv6 ICMP Rate Limiting Configuration

The following example shows an interval of 50 milliseconds and a bucket size of 20 tokens being configured for IPv6 ICMP error messages:

```
ipv6 icmp error-interval 50 20
```

Example: Cisco Express Forwarding and Distributed Cisco Express Forwarding Configuration

In the following example, both Cisco Express Forwarding for IPv6 and network accounting for Cisco Express Forwarding for IPv6 have been enabled globally on a nondistributed architecture device, and Cisco Express Forwarding for IPv6 has been enabled on Ethernet interface 0. The example also shows that the forwarding of IPv6 unicast datagrams has been configured globally on the device with the **ipv6 unicast-routing** command, an IPv6 address has been configured on Ethernet interface 0 with the **ipv6 address** command, and Cisco Express Forwarding for IPv4 has been configured globally on the device with the **ip cef** command.

```
ip cef
ipv6 unicast-routing
ipv6 cef
ipv6 cef accounting prefix-length
interface Ethernet0
 ip address 10.4.9.11 255.0.0.0
 media-type 10BaseT
 ipv6 address 2001:DB8:C18:1::/64 eui-64
```

In the following example, both distributed Cisco Express Forwarding for IPv6 and network accounting for distributed Cisco Express Forwarding for IPv6 have been enabled globally on a distributed architecture device. The forwarding of IPv6 unicast datagrams has been configured globally on the device with the **ipv6 unicast-routing** command and distributed Cisco Express Forwarding for IPv4 has been configured globally on the device with the **ip cef distributed** command.

```
ip cef distributed
ipv6 unicast-routing
ipv6 cef distributed
ipv6 cef accounting prefix-length
```

Example: Hostname-to-Address Mappings Configuration

The following example defines two static hostname-to-address mappings in the hostname cache, establishes a domain list with several alternate domain names to complete unqualified hostnames, specifies host 2001:DB8::250:8bff:fee8:f800 and host 2001:DB8:0:f004::1 as the name servers, and reenables the DNS service:

```
ipv6 host cisco-sj 2001:DB8:700:20:1::12
ipv6 host cisco-hq 2001:DB8:768::1 2001:DB8:20:1::22
ip domain list domain1-list.com
ip domain list serviceprovider2-name.com
ip domain list college2-name.edu
ip name-server 2001:DB8::250:8bff:fee8:f800 2001:DB8:0:f004::1
ip domain-lookup
```

Examples: IPv6 Address to ATM and Frame Relay PVC Mapping Configuration

Example: IPv6 ATM PVC Mapping Configuration (Point-to-Point Interface)

In the following example, two nodes named Router 1 and Router 2 are connected by a single PVC. The point-to-point subinterface ATM0.132 is used on both nodes to terminate the PVC; therefore, the mapping between the IPv6 addresses of both nodes and the PVC is implicit (no additional mappings are required).

Router 1 Configuration

```
interface ATM 0
 no ip address
!
interface ATM 0.132 point-to-point
 pvc 1/32
 encapsulation aal5snap
 !
 ipv6 address 2001:DB8:2222:1003::72/64
```

Router 2 Configuration

```
interface ATM 0
 no ip address
!
interface ATM 0.132 point-to-point
 pvc 1/32
 encapsulation aal5snap
 !
 ipv6 address 2001:DB8:2222:1003::45/64
```

Example: IPv6 ATM PVC Mapping Configuration (Point-to-Multipoint Interface)

In the following example, the same two nodes (Router 1 and Router 2) from the previous example are connected by the same PVC. In this example, however, the point-to-multipoint interface ATM0 is used on

both nodes to terminate the PVC; therefore, explicit mappings are required between the link-local and global IPv6 addresses of interface ATM0 on both nodes and the PVC. Additionally, ATM pseudobroadcasts are enabled on the link-local address of interface ATM0 on both nodes. The link-local address specified here is the link-local address of the other end of the PVC.

Router 1 Configuration

```
interface ATM 0
 no ip address
 pvc 1/32
 protocol ipv6 2001:DB8:2222:1003::45
 protocol ipv6 FE80::60:2FA4:8291:2 broadcast
 encapsulation aal5snap
 !
 ipv6 address 2001:DB8:2222:1003::72/64
```

Router 2 Configuration

```
interface ATM 0
 no ip address
 pvc 1/32
 protocol ipv6 FE80::60:3E47:AC8:C broadcast
 protocol ipv6 2001:DB8:2222:1003::72
 encapsulation aal5snap
 !
 ipv6 address 2001:DB8:2222:1003::45/64
```

Example: IPv6 Frame Relay PVC Mapping Configuration (Point-to-Point Interface)

In the following example, three nodes named Router A, Router B, and Router C make up a fully meshed network. Each node is configured with two PVCs, which provide an individual connection to each of the other two nodes. Each PVC is configured on a different point-to-point subinterface, which creates three unique IPv6 networks (2001:DB8:2222:1017:/64, 2001:DB8:2222:1018::/64, and 2001:DB8:2222:1019::/64). Therefore, the mappings between the IPv6 addresses of each node and the DLCI (DLCI 17, 18, and 19) of the PVC used to reach the addresses are implicit (no additional mappings are required).

Note Given that each PVC in the following example is configured on a different point-to-point subinterface, the configuration in the following example can also be used in a network that is not fully meshed. Additionally, configuring each PVC on a different point-to-point subinterface can help simplify your routing protocol configuration. However, the configuration in the following example requires more than one IPv6 network, whereas configuring each PVC on point-to-multipoint interfaces requires only one IPv6 network.

Router A Configuration

```
interface Serial 3
 encapsulation frame-relay
!
interface Serial3.17 point-to-point
 description to Router B
 ipv6 address 2001:DB8:2222:1017::46/64
 frame-relay interface-dlci 17
!
interface Serial 3.19 point-to-point
 description to Router C
 ipv6 address 2001:DB8:2222:1019::46/64
 frame-relay interface-dlci 19
```

Router B Configuration

```
interface Serial 5
 encapsulation frame-relay
!
interface Serial5.17 point-to-point
 description to Router A
 ipv6 address 2001:DB8:2222:1017::73/64
 frame-relay interface-dlci 17
!
interface Serial5.18 point-to-point
 description to Router C
 ipv6 address 2001:DB8:2222:1018::73/64
 frame-relay interface-dlci 18
```

Router C Configuration

```
interface Serial 0
 encapsulation frame-relay
!
interface Serial0.18 point-to-point
 description to Router B
 ipv6 address 2001:DB8:2222:1018::72/64
 frame-relay interface-dlci 18
!
interface Serial0.19 point-to-point
 description to Router A
 ipv6 address 2001:DB8:2222:1019::72/64
 frame-relay interface-dlci 19
```

Example: IPv6 Frame Relay PVC Mapping Configuration (Point-to-Multipoint Interface)

In the following example, the same three nodes (Router A, Router B, and Router C) from the previous example make up a fully meshed network and each node is configured with two PVCs (which provide an individual connection to each of the other two nodes). However, the two PVCs on each node in the following example are configured on a single interface (serial 3, serial 5, and serial 10, respectively), which makes each interface a point-to-multipoint interface. Therefore, explicit mappings are required between the link-local and global IPv6 addresses of each interface on all three nodes and the DLCI (DLCI 17, 18, and 19) of the PVC used to reach the addresses.

Router A Configuration

```
interface Serial 3
 encapsulation frame-relay
 ipv6 address 2001:DB8:2222:1044::46/64
 frame-relay map ipv6 FE80::E0:F727:E400:A 17 broadcast
 frame-relay map ipv6 FE80::60:3E47:AC8:8 19 broadcast
 frame-relay map ipv6 2001:DB8:2222:1044::72 19
 frame-relay map ipv6 2001:DB8:2222:1044::73 17
```

Router B Configuration

```
interface Serial 5
 encapsulation frame-relay
 ipv6 address 2001:DB8:2222:1044::73/64
 frame-relay map ipv6 FE80::60:3E59:DA78:C 17 broadcast
 frame-relay map ipv6 FE80::60:3E47:AC8:8 18 broadcast
 frame-relay map ipv6 2001:DB8:2222:1044::46 17
 frame-relay map ipv6 2001:DB8:2222:1044::72 18
```

Router C Configuration

```
interface Serial 10
 encapsulation frame-relay
 ipv6 address 2001:DB8:2222:1044::72/64
 frame-relay map ipv6 FE80::60:3E59:DA78:C 19 broadcast
 frame-relay map ipv6 FE80::E0:F727:E400:A 18 broadcast
 frame-relay map ipv6 2001:DB8:2222:1044::46 19
 frame-relay map ipv6 2001:DB8:2222:1044::73 18
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 supported feature list	"Start Here: Cisco IOS Software Release Specifics for IPv6 Features," *Cisco IOS IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*
IPv6 DHCP description and configuration	"Implementing DHCP for IPv6," *Cisco IOS IPv6 Configuration Guide*
IPv4 addressing configuration tasks	"Configuring IPv4 Addresses," *Cisco IOS IP Addressing Services Configuration Guide*
IPv4 services configuration tasks	"Configuring IP Services," *Cisco IOS IP Application Services Configuration Guide*
IPv4 addressing commands	*Cisco IOS IP Addressing Services Command Reference*
IPv4 IP services commands	*Cisco IOS IP Application Services Command Reference*
Stateful switchover	"Stateful Switchover," *Cisco IOS High Availability Configuration Guide*
Switching configuration tasks	*Cisco IOS IP Switching Configuration Guide*
Switching commands	*Cisco IOS IP Switching Command Reference*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIBs	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 1981	*Path MTU Discovery for IP version 6*
RFC 2373	*IP Version 6 Addressing Architecture*
RFC 2374	*An Aggregatable Global Unicast Address Format*
RFC 2460	*Internet Protocol, Version 6 (IPv6) Specification*
RFC 2461	*Neighbor Discovery for IP Version 6 (IPv6)*
RFC 2462	*IPv6 Stateless Address Autoconfiguration*
RFC 2463	*Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification*
RFC 2464	*Transmission of IPv6 Packets over Ethernet Networks*
RFC 2467	*Transmission of IPv6 Packets over FDDI Networks*
RFC 2472	*IP Version 6 over PPP*
RFC 2492	*IPv6 over ATM Networks*
RFC 2590	*Transmission of IPv6 Packets over Frame Relay Networks Specification*
RFC 2684	*Multiprotocol Encapsulation over ATM Adaptation Layer 5*
RFC 3152	*Delegation of IP6.ARPA*
RFC 3162	*RADIUS and IPv6*
RFC 3513	*Internet Protocol Version 6 (IPv6) Addressing Architecture*
RFC 3596	*DNS Extensions to Support IP version 6*
RFC 3879	*Deprecating Site Local Addresses*

RFCs	Title
RFC 4193	*Unique Local IPv6 Unicast Addresses*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing IPv6 Addressing and Basic Connectivity

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 8 Feature Information for Implementing IPv6 Addressing and Basic Connectivity

Feature Name	Releases	Feature Information
IPv6—Anycast Address	12.2(25)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(33)SXH 12.3(4)T 12.4 12.4(2)T	An anycast address is an address that is assigned to a set of interfaces that typically belong to different nodes.
IPv6—Base Protocols High Availability	12.2(33)SRE	IPv6 neighbor discovery supports SSO.

Feature Name	Releases	Feature Information
IPv6—ICMP Rate Limiting	12.2(8)T 12.3 12.3(2)T 12.4 12.4(2)T	The IPv6 ICMP Rate Limiting feature implements a token bucket algorithm for limiting the rate at which IPv6 ICMP error messages are sent out on the network.
IPv6—ICMPv6	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(17a)SX1 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T	ICMP for IPv6 generates error messages, such as ICMP destination unreachable messages, and informational messages, such as ICMP echo request and reply messages. Additionally, ICMP packets in IPv6 are used in the IPv6 neighbor discovery process, path MTU discovery, and the MLD protocol for IPv6.
IPv6—ICMPv6 Redirect	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(17a)SX1 12.2(4)T 12.3 12.3(2)T 12.4 12.4(2)T	A value of 137 in the Type field of the ICMP packet header identifies an IPv6 neighbor redirect message. Routers send neighbor redirect messages to inform hosts of better first-hop nodes on the path to a destination.
IPv6—IPv6 Default Router Preferences	12.2(33)SB 12.2(33)SRA 12.4(2)T 12.2(33)SXH 15.0(1)S	The DRP extension provides a coarse preference metric (low, medium, or high) for default routers.

Feature Name	Releases	Feature Information
IPv6—IPv6 MTU Path Discovery	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(17a)SX1 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	Path MTU discovery in IPv6 allows a host to dynamically discover and adjust to differences in the MTU size of every link along a given data path.
IPv6—IPv6 Neighbor Discovery	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(17a)SX1 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	The IPv6 neighbor discovery process uses ICMP messages and solicited-node multicast addresses to determine the link-layer address of a neighbor on the same network (local link), verify the reachability of a neighbor, and track neighboring routers.

Feature Name	Releases	Feature Information
IPv6—IPv6 Neighbor Discovery Duplicate Address Detection	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(17a)SX1 12.2(4)T 12.3 12.3(2)T 12.4 12.4(2)T	IPv6 neighbor discovery duplicate address detection is performed first on a new, link-local IPv6 address before the address is assigned to an interface (the new address remains in a tentative state while duplicate address detection is performed).
IPv6—IPv6 Stateless Autoconfiguration	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(17a)SX1 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	The IPv6 Stateless Autoconfiguration feature can be used to manage link, subnet, and site addressing changes.
IPv6—IPv6 Static Cache Entry for Neighbor Discovery	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(17a)SX1 12.2(8)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	The IIPv6 Static Cache Entry for Neighbor Discovery feature allows static entries to be made in the IPv6 neighbor cache.

Feature Name	Releases	Feature Information
IPv6—Per-Interface Neighbor Discovery Cache Limit	15.1(3)T	The Per-Interface Neighbor Discovery Cache Limit feature provides the ability to limit the number of neighbor discovery cache entries on a per interface basis. The following commands were introduced or modified for this feature: **ipv6 nd cache interface-limit (global)**, **ipv6 nd cache interface-limit (interface)**, **show ipv6 neighbors**.
IPv6 Access Services: Routed Bridged Encapsulation (RBE)	12.3(4)T 12.4 12.4(2)T	RBE provides a mechanism for routing a protocol from a bridged interface to another routed or bridged interface.
IPv6 Address Types—Unicast	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(17a)SX1 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T	An IPv6 unicast address is an identifier for a single interface, on a single node.
IPv6 Data Link—ATM PVC and ATM LANE	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T	In IPv6 networks, a data link is a network sharing a particular link-local prefix. ATM PVC and ATM LANE are data links supported for IPv6.

Feature Name	Releases	Feature Information
IPv6 Data Link—Cisco High-Level Data Link Control (HDLC)	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T	In IPv6 networks, a data link is a network sharing a particular link-local prefix. HDLC is a type of data link supported for IPv6.
IPv6 Data Link—Dynamic Packet Transport (DPT)	12.0(23)S	In IPv6 networks, a data link is a network sharing a particular link-local prefix. DPT is a type of data link supported for IPv6.
IPv6 Data Link—Ethernet, Fast Ethernet, Gigabit Ethernet, and 10-Gigabit Ethernet	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T	In IPv6 networks, a data link is a network sharing a particular link-local prefix. Ethernet, Fast Ethernet, Gigabit Ethernet, and 10-Gigabit Ethernet are data links supported for IPv6.
IPv6 Data Link—FDDI	12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T	In IPv6 networks, a data link is a network sharing a particular link-local prefix. FDDI is a type of data link supported for IPv6.

Feature Name	Releases	Feature Information
IPv6 Data Link—Frame Relay PVC	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T	In IPv6 networks, a data link is a network sharing a particular link-local prefix. Frame relay PVC is a type of data link supported for IPv6.
IPv6 Data Link—PPP service over Packet over SONET, ISDN, and serial (synchronous and asynchronous) interfaces	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T	In IPv6 networks, a data link is a network sharing a particular link-local prefix. PPP service over Packet over SONET, ISDN, and serial interfaces is a type of data link supported for IPv6.
IPv6 Data Link—VLANs using Cisco Inter-Switch Link (ISL)	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(18)SXE 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	In IPv6 networks, a data link is a network sharing a particular link-local prefix. VLANs using Cisco ISL is a type of data link supported for IPv6.

Feature Name	Releases	Feature Information
IPv6 Data Link—VLANs using IEEE 802.1Q encapsulation	12.0(22)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(18)SXE 12.2(14)S 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	In IPv6 networks, a data link is a network sharing a particular link-local prefix. VLANs using IEEE 802.1Q encapsulation is a type of data link supported for IPv6.
Enhanced IPv6 Neighbor Discovery Cache Management	12.2(33)SXI7 15.0(1)SY1	The IPv6 highly scalable neighbor discovery feature optimizes IPv6 neighbor discovery by providing ND cache autorefresh, unsolicited NA gleaning, and NUD exponential retransmit.
IPv6 Services—AAAA DNS lookups over an IPv4 Transport	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(17a)SX1 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	IPv6 basic connectivity can be enhanced by configuring support for AAAA record types in the DNS name-to-address and address-to-name lookup processes.

Feature Name	Releases	Feature Information
IPv6 Services—Cisco Discovery Protocol--IPv6 Address Family Support for Neighbor Information	12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(18)SXE 12.2(8)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	The Cisco Discovery Protocol IPv6 Address Support for Neighbor Information feature adds the ability to transfer IPv6 addressing information between two Cisco devices.
IPv6 Services—DNS Lookups over an IPv6 Transport	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRE2 12.2(8)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	IPv6 supports DNS record types that are supported in the DNS name-to-address and address-to-name lookup processes.
IPv6 Services—Generic Prefix	12.3(4)T 12.4 12.4(2)T	The upper 64 bits of an IPv6 address are composed from a global routing prefix plus a subnet ID. A general prefix (for example, /48) holds a short prefix, based on which a number of longer, more specific, prefixes (for example, /64) can be defined.

Feature Name	Releases	Feature Information
IPv6 Switching—Cisco Express Forwarding and Distributed Cisco Express Forwarding Support	12.0(21)ST 12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(17a)SX1 12.2(13)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	Cisco Express Forwarding for IPv6 is advanced, Layer 3 IP switching technology for the forwarding of IPv6 packets. Distributed Cisco Express Forwarding for IPv6 performs the same functions as CEFv6 but for distributed architecture platforms such as the GSRs and the Cisco 7500 series routers.
IPv6 Support on BVI Interfaces	15.1(2)T	This feature allows IPv6 commands to be supported on BVI so that users can assign IPv6 addresses to a BVI and route IPv6 packets.
Unicast Reverse Path Forwarding for IPv6	12.0(31)S 12.2(50)SY	The Unicast RPF feature mitigates problems caused by malformed or forged (spoofed) IPv6 source addresses that pass through an IPv6 router. The following command was introduced: **ipv6 verify unicast source reachable-via**.

Implementing ADSL and Deploying Dial Access for IPv6

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Implementing ADSL and Deploying Dial Access for IPv6

- ADSL and dial deployment are available for interfaces with PPP encapsulation enabled, including PPP over ATM (PPPoA), PPP over Ethernet (PPPoE), PPP over async, and PPP over ISDN.
- Network Address Translation (NAT) is not supported for IPv6 TACACS servers in Cisco IOS Release 15.1(1)S.

Information About Implementing ADSL and Deploying Dial Access for IPv6

Address Assignment for IPv6

A Cisco router configured with IPv6 will advertise its IPv6 prefixes on one or more interfaces, allowing IPv6 clients to automatically configure their addresses. In IPv6, address assignment is performed at the network layer, in contrast to IPv4 where a number of functions are handled in the PPP layer. The only function handled in IPv6 control protocol is the negotiation of a unique interface identifier. Everything else, including Domain Name Server (DNS) server discovery, is done within the IPv6 protocol itself.

Contrary to IPv4 address assignment, an IPv6 user will be assigned a prefix, not a single address. Typically, the ISP assigns a 64- or 48-bit prefix.

In IPv6, ISPs assign long-lived prefixes to users, which has some impact on the routing system. In typical IPv4 environments, each network access server (NAS) has a pool of 24-bit addresses and users get addresses from this pool when dialing in. If a user dials another point of presence (POP) or is connected to another NAS at the same POP, a different IPv4 address is assigned.

Addresses for IPv6 are assigned using two methods:

Stateless Address Autoconfiguration

Assigning addresses using the stateless address autoconfiguration method can be used only to assign 64-bit prefixes. Each user is assigned a 64-bit prefix, which is advertised to the user in a router advertisement (RA). All addresses are automatically configured based on the assigned prefix.

A typical scenario is to assign a separate 64-bit prefix per user; however, users can also be assigned a prefix from a shared pool of addresses. Using the shared pool limits addresses to only one address per user.

This method works best for the cases where the customer provider edge (CPE) router is a single PC or is limited to only one subnet. If the user has multiple subnets, Layer 2 (L2) bridging, multilink subnets or proxy RA can be used. The prefix advertised in the RA can come from an authorization, authentication, and accounting (AAA) server, which also provides the prefix attribute, can be manually configured, or can be allocated from a prefix pool.

The Framed-Interface-Id AAA attribute influences the choice of interface identifier for peers and, in combination with the prefix, the complete IPv6 address can be determined.

Prefix Delegation

Prefix delegation uses Dynamic Host Configuration Protocol (DHCP). When the user requests a prefix from the prefix delegator, typically the NAS, the prefix is allocated.

An IPv6 prefix delegating router selects IPv6 prefixes to be assigned to a requesting router upon receiving a request from the client. The delegating router might select prefixes for a requesting router in the following ways:

- Static assignment based on subscription to an ISP
- Dynamic assignment from a pool of available prefixes
- Selection based on an external authority such as a RADIUS server using the Framed-IPv6-Prefix attribute.

DHCP SIP Server Options

Two DHCP for IPv6 Session Initiation Protocol (SIP) server options describe a local outbound SIP proxy: one carries a list of domain names, the other a list of IPv6 addresses. These two options can be configured in a DHCPv6 configuration pool.

AAA over IPv6

Vendor-specific attributes (VSAs) are used to support AAA over IPv6. Cisco VSAs are inacl, outacl, prefix, and route.

You can configure prefix pools and pool names by using the AAA protocol. Customers can deploy an IPv6 RADIUS server or a TACACS+ server to communicate with Cisco devices.

RADIUS over IPv6

The following RADIUS attributes, as described in RFC 3162, are supported for IPv6:

- Framed-Interface-Id
- Framed-IPv6-Pool
- Framed-IPv6-Prefix
- Framed-IPv6-Route
- Login-IPv6-Host

The following RADIUS attributes are also supported for IPv6:

- Delegated-IPv6-Prefix (RFC 4818)
- Delegated-IPv6-Prefix-Pool
- DNS-Server-IPv6-Address
- IPv6 ACL
- IPv6_DNS_Servers
- IPv6 Pool
- IPv6 Prefix#
- IPv6 Route

The attributes listed above can be configured on a RADIUS server and downloaded to access servers, where they can be applied to access connections.

RADIUS Per-User Attributes for Virtual Access in IPv6 Environments

The following IPv6 attributes for RADIUS attribute-value (AV) pairs are supported for virtual access:

Framed-Interface-Id

The Framed-Interface-Id attribute indicates the IPv6 interface identifier to be configured. This per-user attribute is used during the IPv6CP negotiations and may be used in access-accept packets. If the Interface-

Identifier IPv6CP option has been successfully negotiated, this attribute must be included in an Acc-0Request packet as a hint by the NAS to the server that it would prefer that value.

Framed-IPv6-Pool

The Framed-IPv6-Pool attribute is a per-user attribute that contains the name of an assigned pool that should be used to assign an IPv6 prefix for the user. This pool should either be defined locally on the router or defined on a RADIUS server from which pools can be downloaded.

Framed-IPv6-Prefix

The Framed-IPv6-Prefix attribute performs the same function as the Cisco VSA--it is used for virtual access only and indicates an IPv6 prefix (and corresponding route) to be configured. This attribute is a per-user attribute and lets the user specify which prefixes to advertise in Neighbor Discovery Router Advertisement messages. The Framed-IPv6-Prefix attribute may be used in access-accept packets and can appear multiple times. The NAS will create a corresponding route for the prefix.

To use this attribute for DHCP for IPv6 prefix delegation, create a profile for the same user on the RADIUS server. The username associated with the second profile has the suffix "-dhcpv6."

The Framed-IPv6-Prefix attribute in the two profiles is treated differently. If a NAS needs both to send a prefix in router advertisements (RAs) and delegate a prefix to a remote user’s network, the prefix for RA is placed in the Framed-IPv6-Prefix attribute in the user’s regular profile, and the prefix used for prefix delegation is placed in the attribute in the user’s separate profile.

Framed-IPv6-Route

The Framed-IPv6-Route attribute performs the same function as the Cisco VSA: It is a per-user attribute that provides routing information to be configured for the user on the NAS. This attribute is a string attribute and is specified using the **ipv6 route** command.

IPv6 ACL

You can specify a complete IPv6 access list. The unique name of the access list is generated automatically. The access list is removed when its user logs out. The previous access list on the interface is reapplied.

The inacl and outacl attributes allow you to a specific existing access list configured on the router. The following example shows ACL number 1 specified as the access list:

```
cisco-avpair = "ipv6:inacl#1=permit 2001:DB8:cc00:1::/48",
cisco-avpair = "ipv6:outacl#1=deny 2001:DB8::/10",
```

IPv6 Pool

For RADIUS authentication, the IPv6 Pool attribute extends the IPv4 address pool attributed to support the IPv6 protocol. It specifies the name of a local pool on the NAS from which to get the prefix and is used whenever the service is configured as PPP and whenever the protocol is specified as IPv6. Note that the address pool works in conjunction with local pooling. It specifies the name of the local pool that has been preconfigured on the NAS.

IPv6 Prefix

The IPv6 Prefix# attribute lets you indicate which prefixes to advertise in Neighbor Discovery Router Advertisement messages. When the IPv6 Prefix# attribute is used, a corresponding route (marked as a per-user static route) is installed in the routing information base (RIB) tables for the given prefix.

```
cisco-avpair = "ipv6:prefix#1=2001:DB8::/64",
cisco-avpair = "ipv6:prefix#2=2001:DB8::/64",
```

IPv6 Route

The IPv6 route attribute allows you to specify a per-user static route. A static route is appropriate when the Cisco IOS software cannot dynamically build a route to the destination. See the description of the **ipv6 route** command for more information about building static routes.

The following example shows the IPv6 route attribute used to define a static route:

```
cisco-avpair = "ipv6:route#1=2001:DB8:cc00:1::/48",
cisco-avpair = "ipv6:route#2=2001:DB8:cc00:2::/48",
```

Login-IPv6-Host

The Login-IPv6-Host attribute is a per-user attribute that indicates the IPv6 system with which to connect the user when the Login-Service attribute is included.

TACACS+ Over an IPv6 Transport

An IPv6 server can be configured to use TACACS+. Both IPv6 and IPv4 servers can be configured to use TACACS+ using a name instead of an IPv4 or IPv6 address.

IPv6 Prefix Pools

The function of prefix pools in IPv6 is similar to that of address pools in IPv4. The main difference is that IPv6 assigns prefixes rather than single addresses.

As in IPv4, a pool or a pool definition in IPv6 can be configured locally or it can be retrieved from an AAA server. Overlapping membership between pools is not permitted.

Once a pool is configured, it cannot be changed. If you change the configuration, the pool will be removed and re-created. All prefixes previously allocated will be freed.

Prefix pools can be defined so that each user is allocated a 64-bit prefix or so that a single prefix is shared among several users. In a shared prefix pool, each user may receive only one address from the pool.

How to Configure ADSL and Deploy Dial Access in IPv6

- Configuring the NAS, page 114
- Configuring the Remote CE Router, page 117
- Configuring the DHCPv6 Server to Obtain Prefixes from RADIUS Servers, page 119
- Configuring DHCPv6 AAA and SIP Options, page 120
- Configuring TACACS+ over IPv6, page 121

Configuring the NAS

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **hostname** *name*
4. **aaa new-model**
5. **aaa authentication ppp** {**default** | *list-name*} *method1* [*method2*...]
6. **aaa authorization configuration default** {**radius** | **tacacs+**
7. **show ipv6 route** [*ipv6-address* | *ipv6-prefix* / *prefix-length* | *protocol* | *interface-type interface-number*
8. **virtual-profile virtual-template** *number*
9. **interface serial** *controller-number* **:** *timeslot*
10. **encapsulation** *encapsulation-type*
11. **exit**
12. **dialer-group** *group-number*
13. **ppp authentication** *protocol1* [*protocol2*...] [**if-needed**] [*list-name* | **default**] [**callin**] [**one-time**] [**optional**]
14. **interface virtual-template** *number*
15. **ipv6 enable**
16. **dialer-list** *dialer-group* **protocol** *protocol-name* {**permit** | **deny** | **list** *access-list-number* | *access-group*}
17. **radius-server host** {*hostname* | *ip-address*} [**test username** *user-name*] [**auth-port** *port-number*] [**ignore-auth-port**] [**acct-port** *port-number*] [**ignore-acct-port**] [**timeout** *seconds*] [**retransmit** *retries*] [**key** *string*] [**alias** {*hostname* | *ip-address*}] [**idle-time** *seconds*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**hostname** *name* **Example:** `Router(config)# hostname cust1-53a`	Specifies the hostname for the network server.
Step 4	**aaa new-model** **Example:** `Router(config)# aaa new-model`	Enables the AAA server.
Step 5	**aaa authentication ppp** {**default** | *list-name*} *method1* [*method2...*] **Example:** `Router(config)# aaa authentication ppp default if-needed group radius`	Specifies one or more AAA authentication methods for use on serial interfaces that are running PPP.
Step 6	**aaa authorization configuration default** {**radius** | **tacacs+** **Example:** `Router(config)# aaa authorization configuration default radius`	Downloads configuration information from the AAA server.
Step 7	**show ipv6 route** [*ipv6-address* | *ipv6-prefix* / *prefix-length* | *protocol* | *interface-type interface-number* **Example:** `Router(config)# show ipv6 route`	Shows the routes installed by the previous commands.
Step 8	**virtual-profile virtual-template** *number* **Example:** `Router(config)# virtual-profile virtual-template 1`	Enables virtual profiles by virtual interface template.
Step 9	**interface serial** *controller-number* **:** *timeslot* **Example:** `Router(config)# interface serial 0:15`	Specifies a serial interface created on a channelized E1 or channelized T1 controller (for ISDN PRI, channel-associated signaling, or robbed-bit signaling). This command also puts the router into interface configuration mode.

Command or Action	Purpose
Step 10 **encapsulation** *encapsulation-type* **Example:** `Router(config-if)# encapsulation ppp`	Sets the encapsulation method used by the interface.
Step 11 **exit** **Example:** `Router(config-if)# exit`	Returns to global configuration mode.
Step 12 **dialer-group** *group-number* **Example:** `Router(config)# dialer-group 1`	Controls access by configuring an interface to belong to a specific dialing group.
Step 13 **ppp authentication** *protocol1* [*protocol2*...] [**if-needed**] [*list-name* | **default**] [**callin**] [**one-time**] [**optional**] **Example:** `Router(config)# ppp authentication chap`	Enables Challenge Handshake Authentication Protocol (CHAP) or Password Authentication Protocol (PAP) or both and specifies the order in which CHAP and PAP authentication are selected on the interface.
Step 14 **interface virtual-template** *number* **Example:** `Router(config)# interface virtual-template 1`	Creates a virtual template interface that can be configured and applied dynamically in creating virtual access interfaces.
Step 15 **ipv6 enable** **Example:** `Router(config)# ipv6 enable`	Enables IPv6 processing on an interface that has not been configured with an explicit IPv6 address.
Step 16 **dialer-list** *dialer-group* **protocol** *protocol-name* {**permit** | **deny** | **list** *access-list-number* | *access-group*} **Example:** `Router(config)# dialer-list 1 protocol ipv6 permit`	Defines a dial-on-demand routing (DDR) dialer list for dialing by protocol or by a combination of a protocol and a previously defined access list.

	Command or Action	Purpose
Step 17	**radius-server host** {*hostname* | *ip-address*} [**test username** *user-name*] [**auth-port** *port-number*] [**ignore-auth-port**] [**acct-port** *port-number*] [**ignore-acct-port**] [**timeout** *seconds*] [**retransmit** *retries*] [**key** *string*] [**alias** {*hostname* | *ip-address*}] [**idle-time** *seconds* **Example:** `Router(config)# radius-server host 172.17.250.8 auth-port 1812 acct-port 1813 key testing123`	Specifies a RADIUS server host.

Configuring the Remote CE Router

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **hostname** *name*
4. **interface bri** *number* **.** *subinterface-number* [**multipoint** | **point-to-point**]
5. **encapsulation** *encapsulation-type*
6. **ipv6 address autoconfig** [**default**
7. **isdn switch-type** *switch-type*
8. **ppp authentication** {*protocol1* [*protocol2...*]} [**if-needed**] [*list-name* | **default**] [**callin**] [**one-time**]
9. **ppp multilink** [**bap** | **required**]
10. **exit**
11. **dialer-list** *dialer-group* **protocol** *protocol-name* {**permit** | **deny** | **list** *access-list-number* | *access-group*}
12. **ipv6 route** *ipv6-prefix / prefix-length* {*ipv6-address* | *interface-type interface-number ipv6-address*]} [*administrative-distance*] [*administrative-multicast-distance* | **unicast**| **multicast**] [**tag** *tag*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**hostname** *name* **Example:** `Router(config)# hostname cust1-36a`	Specifies the hostname for the network server.
Step 4	**interface bri** *number* **.** *subinterface-number* [**multipoint** | **point-to-point**] **Example:** `Router(config)# interface bri 1.0`	Configures a BRI interface.
Step 5	**encapsulation** *encapsulation-type* **Example:** `Router(config-if)# encapsulation ppp`	Sets the encapsulation method used by the interface.
Step 6	**ipv6 address autoconfig** [**default** **Example:** `Router(config-if)# ipv6 address autoconfig`	Indicates that the IPv6 address will be generated automatically.
Step 7	**isdn switch-type** *switch-type* **Example:** `Router(config-if)# isdn switch-type basic-net3`	Specifies the central office switch type on the ISDN interface.
Step 8	**ppp authentication** {*protocol1* [*protocol2*...]} [**if-needed**] [*list-name* | **default**] [**callin**] [**one-time**] **Example:** `Router(config-if)# ppp authentication chap`	Enables Challenge Handshake Authentication Protocol (CHAP) or Password Authentication Protocol (PAP) or both and specifies the order in which CHAP and PAP authentication are selected on the interface.

	Command or Action	Purpose
Step 9	**ppp multilink** [**bap** | **required**] **Example:** `Router(config-if)# ppp multilink`	Enables Multilink PPP (MLP) on an interface and, optionally, enables Bandwidth Allocation Control Protocol (BACP) and Bandwidth Allocation Protocol (BAP) for dynamic bandwidth allocation.
Step 10	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 11	**dialer-list** *dialer-group* **protocol** *protocol-name* {**permit** | **deny** | **list** *access-list-number* | *access-group*} **Example:** `Router(config)# dialer-list 1 protocol ipv6 permit`	Defines a dial-on-demand routing (DDR) dialer list for dialing by protocol or by a combination of a protocol and a previously defined access list.
Step 12	**ipv6 route** *ipv6-prefix / prefix-length* {*ipv6-address* | *interface-type interface-number ipv6-address*]} [*administrative-distance*] [*administrative-multicast-distance* | **unicast**| **multicast**] [**tag** *tag* **Example:** `Router(config)# ipv6 route 2001:DB8::1/128 BRI1/0`	Establishes static IPv6 routes. • Use one command for each route.

Configuring the DHCPv6 Server to Obtain Prefixes from RADIUS Servers

Before you perform this task, you must configure the AAA client and PPP on the router.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 nd prefix framed-ipv6-prefix**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface ethernet 0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**ipv6 nd prefix framed-ipv6-prefix** **Example:** `Router(config-if)# ipv6 nd prefix framed-ipv6-prefix`	Adds the prefix in a received RADIUS framed IPv6 prefix attribute to the interface's neighbor discovery prefix queue.

Configuring DHCPv6 AAA and SIP Options

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **prefix-delegation aaa** [**method-list** *method-list*] [*lifetime*]
5. **sip address** *ipv6-address*
6. **sip domain-name** *domain-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Router(config)# ipv6 dhcp pool pool1`	Configures a DHCP for IPv6 configuration information pool and enters DHCP for IPv6 pool configuration mode.
Step 4	**prefix-delegation aaa** [**method-list** *method-list*] [*lifetime*] **Example:** `Router(config-dhcp)# prefix-delegation aaa method-list list1`	Specifies that prefixes are to be acquired from AAA servers.
Step 5	**sip address** *ipv6-address* **Example:** `Router(config-dhcp)# sip address 2001:DB8::2`	Configures a SIP server IPv6 address to be returned in the SIP server's IPv6 address list option to clients.
Step 6	**sip domain-name** *domain-name* **Example:** `Router(config-dhcp)# sip domain sip1.cisco.com`	Configures a SIP server domain name to be returned in the SIP server's domain name list option to clients.

Configuring TACACS+ over IPv6

Configuring the TACACS+ Server over IPv6

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **tacacs server** *name*
4. **address ipv6** *ipv6-address*
5. **key** [**0** | **7**] *key-string*
6. **port** [*number*
7. **send-nat-address**
8. **single-connection**
9. **timeout** *seconds*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**tacacs server** *name* **Example:** `Router(config)# tacacs server server1`	Configures the TACACS+ server for IPv6 and enters TACACS+ server configuration mode.
Step 4	**address ipv6** *ipv6-address* **Example:** `Router(config-server-tacacs)# address ipv6 2001:DB8:3333:4::5`	Configures the IPv6 address of the TACACS+ server.

	Command or Action	Purpose
Step 5	**key** [**0** | **7**] *key-string* **Example:** `Router(config-server-tacacs)# key 0 key1`	Configures the per-server encryption key on the TACACS+ server.
Step 6	**port** [*number* **Example:** `Router(config-server-tacacs)# port 12`	Specifies the TCP port to be used for TACACS+ connections.
Step 7	**send-nat-address** **Example:** `Router(config-server-tacacs)# send-nat-address`	Sends a client's post-NAT address to the TACACS+ server.
Step 8	**single-connection** **Example:** `Router(config-server-tacacs)# single-connection`	Enables all TACACS packets to be sent to the same server using a single TCP connection.
Step 9	**timeout** *seconds* **Example:** `Router(config-server-tacacs)# timeout 10`	Configures the time to wait for a reply from the specified TACACS server.

Specifying the Source Address in TACACS+ Packets

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 tacacs source-interface** *type number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 tacacs source-interface** *type number* **Example:** `Router(config)# ipv6 tacacs source-interface GigabitEthernet 0/0/0`	Specifies an interface to use for the source address in TACACS+ packets.

Configuring TACACS+ Server Group Options

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **aaa group server tacacs+** *group-name*
4. **server name** *server-name*
5. **server-private** {*ip-address* | *name* | *ipv6-address*} [**nat**] [**single-connection**] [**port** *port-number*] [**timeout** *seconds*] [**key** [**0** | **7**] *string*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**aaa group server tacacs+** *group-name* **Example:** `Router(config)# aaa group server tacacs+ group1`	Groups different TACACS+ server hosts into distinct lists and distinct methods.
Step 4	**server name** *server-name* **Example:** `Router(config-sg-tacacs+)# server name server1`	Specifies an IPv6 TACACS+ server.
Step 5	**server-private** {*ip-address* | *name* | *ipv6-address*} [**nat**] [**single-connection**] [**port** *port-number*] [**timeout** *seconds*] [**key** [**0** | **7**] *string*] **Example:** `Router(config-sg-tacacs+)# server-private 2001:DB8:3333:4::5 port 19 key key1`	Configures the IPv6 address of the private TACACS+ server for the group server.

Configuration Examples for Implementing ADSL and Deploying Dial Access for IPv6

Example Implementing ADSL and Deploying Dial Access for IPv6

NAS Configuration

This configuration for the ISP NAS shows the configuration that supports access from the remote CE router.

```
hostname cust1-53a
  aaa new-model
  aaa authentication ppp default if-needed group radius
  aaa authorization network default group radius
  virtual-profile virtual-template 1
  interface Serial0:15
   encapsulation ppp
```

```
 dialer-group 1
 ppp authentication chap
!
interface Virtual-Template1
 ipv6 enable
!
dialer-list 1 protocol ipv6 permit
radius-server host 172.17.250.8 auth-port 1812 acct-port 1813 key testing123
```

Remote CE Router Configuration

This configuration for the remote customer edge router shows PPP encapsulation and IPv6 routes defined.

```
hostname cust-36a
  interface BRI1/0
   encapsulation ppp
   ipv6 enable
   isdn switch-type basic-net3
   ppp authentication chap optional
   ppp multilink
  !
  dialer-list 1 protocol ipv6 permit
  ipv6 route 2001:DB8::1/128 BRI1/0
  ipv6 route ::/0 2001:DB8::1
```

RADIUS Configuration

This RADIUS configuration shows the definition of AV pairs to establish the static routes.

```
  campus1 Auth-Type = Local, Password = mypassword
               User-Service-Type = Framed-User,
               Framed-Protocol = PPP,
               cisco-avpair = ipv6:inacl#1=permit 2001:DB8:0::/64 any,
               cisco-avpair = ipv6:route=2001:DB8:1::/64,
               cisco-avpair = ipv6:route=2001:DB8:2::/64,
               cisco-avpair = ipv6:prefix=2001:DB8:1::/64 0 0 onlink autoconfig,
               cisco-avpair = ipv6:prefix=2001:DB8:2::/64 0 0 onlink autoconfig,
               cisco-avpair = ip:route=10.0.0.0 255.0.0.0,
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 supported feature list	“Start Here: Cisco IOS Software Release Specifics for IPv6 Features ,” *Cisco IOS IPv6 Configuration Guide*
IPv6 basic connectivity	“Implementing IPv6 Addressing and Basic Connectivity, ” *Cisco IOS IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*

Related Topic	Document Title
Certification authority and interoperability, RA proxy	"Security Overview ," *Cisco IOS Security Configuration Guide*
RADIUS server configuration	"Security Overview ," *Cisco IOS Security Configuration Guide*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIBs	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 3162	*RADIUS and IPv6*
RFC 3177	*IAB/IESG Recommendations on IPv6 Address*
RFC 3319	*Dynamic Host Configuration Protocol (DHCPv6) Options for Session Initiated Protocol (SIP) Servers*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing ADSL and Deploying Dial Access for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 9 Feature Information for Implementing ADSL and Deploying Dial Access for IPv6

Feature Name	Releases	Feature Information
AAA Support for Cisco VSA IPv6 Attributes	12.2(13)T 12.3 12.3(2)T 12.4 12.4(2)T	Vendor-specific attributes (VSAs) were developed to support AAA for IPv6.
AAA Support for RFC 3162 IPv6 RADIUS Attributes	12.3(4)T 12.4 12.4(2)T	The AAA attributes for IPv6 are compliant with RFC 3162 and require a RADIUS server capable of supporting RFC 3162.
DHCP for IPv6 Prefix Delegation via AAA	12.2(18)SXE 12.3(14)T 12.4 12.4(2)T	
Enhanced IPv6 Features for ADSL and Dial Deployment	12.2(13)T 12.3 12.3(2)T 12.4 12.4(2)T	Several features were enhanced to enable IPv6 to use ADSL and dial deployment.
IPv6 Prefix Pools	12.2(13)T 12.3 12.3(2)T 12.4 12.4(2)T	The function of prefix pools in IPv6 is similar to that of address pools in IPv4. The main difference is that IPv6 assigns prefixes rather than single addresses.
PPPoA	12.2(13)T 12.3 12.3(2)T 12.4 12.4(2)T	ADSL and dial deployment is available for interfaces with PPP encapsulation enabled, including PPPoA.
PPPoE	12.2(13)T 12.3 12.3(2)T 12.4 12.4(2)T	ADSL and dial deployment is available for interfaces with PPP encapsulation enabled, including PPPoE.
RADIUS over IPv6	12.2(58)SE 15.2(1)T	This feature is supported.

Feature Name	Releases	Feature Information
SSO - PPPoE IPv6	12.2(33)XNE	This feature is supported in Cisco IOS Release 12.2(33)XNE.
TACACS+ over IPv6	12.2(33)SXJ 12.2(58)SE 15.1(1)S 15.2(1)T	TACACS+ over IPv6 is supported. The following commands were introduced or modified by this feature: **aaa group server tacacs +**, **address ipv6 (TACACS+)**, **ipv6 tacacs source-interface**, **key (TACACS+)**, **port (TACACS+)**, **send-nat-address**, **server name (IPv6 TACACS+)**, **server-private (TACACS+)**, **single-connection**, **tacacs server**, **timeout (TACACS+)**.

Implementing Bidirectional Forwarding Detection for IPv6

This document describes how to implement the Bidirectional Forwarding Detection for IPv6 (BFDv6) protocol. BFD is a detection protocol designed to provide fast forwarding path failure detection times for all media types, encapsulations, topologies, and routing protocols. In addition to fast forwarding path failure detection, BFD provides a consistent failure detection method for network administrators. BFDv6 provides IPv6 support by accommodating IPv6 addresses, and it provides the ability to create BFDv6 sessions.

Because the network administrator can use BFD to detect forwarding path failures at a uniform rate, rather than the variable rates for different routing protocol hello mechanisms, network profiling and planning will be easier, and reconvergence time will be consistent and predictable.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Implementing Bidirectional Forwarding Detection for IPv6

IPv6 Cisco Express Forwarding and IPv6 unicast routing must be enabled on all participating routers.

Restrictions for Implementing Bidirectional Forwarding Detection for IPv6

- BFDv6 supports only global IPv6 neighbor addresses if a global IPv6 address is configured on the interface.
- Only asynchronous mode is supported. In asynchronous mode, either BFDv6 peer can initiate a BFDv6 session.

Information About Implementing Bidirectional Forwarding Detection for IPv6

Overview of the BFDv6 Protocol

This section describes the BFDv6 protocol, how it is different from BFD for IPv4, and how it works with BFD for IPv4. BFD is a detection protocol designed to provide fast forwarding path failure detection times for all media types, encapsulations, topologies, and routing protocols. In addition to fast forwarding path failure detection, BFD provides a consistent failure detection method for network administrators. BFDv6 provides IPv6 support by accommodating IPv6 addresses and provides the ability to create BFDv6 sessions.

BFDv6 Registration

BFD clients register with BFD using a registry application program interface (API). The registry arguments include protocol type and the address and interface description block (IDB) of the route to be monitored. These APIs and arguments are all assumed by BFD to be IPv4.

BFDv6 has registries from which these arguments have been removed, and the protocol and encapsulation are described within a session information structure. These session information structures are defined by BFDv6 for the protocols supported. BFDv6 uses information from the session information structures to determine the correct encapsulation for BFDv6 packets on that session.

BFDv6 Global and Link-Local Addresses

BFDv6 supports both global and link-local IPv6 addresses for neighbor creation. BFDv6 sessions select source addresses to match the neighbor address types (for example, global IPv6 address neighbors must be paired with global IPv6 source addresses and link-local IPv6 address neighbors must be paired with link-local IPv6 source addresses). The table below shows the address pairings that BFDv6 supports.

Table 10 BFDv6 Address Pairings for Neighbor Creation

Source Address	Destination Address	Status
Global	Global	Supported
Global	Link local	Not supported
Link local	Global	Not supported
Link local	Link local	Supported

Because all IPv6-enabled interfaces have a link-local address and BFDv6 selects the source address, link-local address neighbors are always paired with a link-local interface address. The link-local source address with global destination address is not supported by Cisco Express Forwarding. Therefore, a global IPv6 address must be configured on an interface before a session with a global address neighbor may be established in BFDv6. BFDv6 rejects any sessions in which the neighbor address is global and no global address is configured on the interface.

Note The behavior of a unique local address (ULA) in BFDv6 is the same as a global address.

BFD for IPv4 and IPv6 on the Same Interface

BFD supports multiple IPv4 and IPv6 sessions per interface, with no restriction on the protocol of those sessions.

Static Route Support for BFD over IPv6

Using the BFDv6 protocol to reach the static route next hop ensures that an IPv6 static route is inserted only in the IPv6 Routing Information Base (RIB) when the next-hop neighbor is reachable. Using the BFDv6 protocol also can remove the IPv6 static route from the IPv6 RIB when the next hop becomes unreachable.

You can configure IPv6 static BFDv6 neighbors. These neighbors can operate in one of two modes: associated (which is the default) and unassociated. A neighbor can be transitioned between the two modes without interrupting the BFDv6 session associated with the neighbor.

- BFDv6 Associated Mode, page 133
- BFDv6 Unassociated Mode, page 134

BFDv6 Associated Mode

In Bidirectional Forwarding Detection for IPv6 (BFDv6) associated mode, an IPv6 static route is automatically associated with an IPv6 static BFDv6 neighbor if the static route next hop exactly matches the static BFDv6 neighbor.

An IPv6 static route requests a BFDv6 session for each static BFDv6 neighbor that has one or more associated IPv6 static routes and is configured over an interface on which BFD has been configured. The state of the BFDv6 session will be used to determine whether the associated IPv6 static routes are inserted in the IPv6 RIB. For example, static routes are inserted in the IPv6 RIB only if the BFDv6 neighbor is

reachable, and the static route is removed from the IPv6 RIB if the BFDv6 neighbor subsequently becomes unreachable.

BFDv6 associated mode requires you to configure a BFD neighbor and static route on both the router on which the BFD-monitored static route is required and on the neighboring router.

BFDv6 Unassociated Mode

An IPv6 static BFD neighbor may be configured as unassociated. In this mode, the neighbor is not associated with static routes, and the neighbor always requests a BFDv6 session if the interface has been configured for BFDv6.

Unassociated mode is useful in the following situations:

- Bringing up a BFDv6 session in the absence of an IPv6 static route—This case occurs when a static route is on router A, with router B as the next hop. Associated mode requires you to create both a static BFD neighbor and static route on both routers in order to bring up the BFDv6 session from B to A. Specifying the static BFD neighbor in unassociated mode on router B avoids the need to configure an unwanted static route.
- Transition to BFD monitoring of a static route—This case occurs when existing IPv6 static routes are inserted in the IPv6 RIB. Here, you want to enable BFD monitoring for these static routes without any interruption to traffic. If you configure an attached IPv6 static BFD neighbor, then the static routes will immediately be associated with the new static BFD neighbor. However, because a static BFD neighbor starts in a down state, the associated static routes are then removed from the IPv6 RIB and are reinserted when the BFDv6 session comes up. Therefore, you will see an interruption in traffic. This interruption can be avoided by configuring the static BFD neighbor as unassociated, waiting until the BFDv6 session has come up, and then reconfiguring the static BFD neighbor as associated.
- Transition from BFD monitoring of a static route—In this case, IPv6 static routes are monitored by BFD and inserted in the RIB. Here, you want to disable BFD monitoring of the static routes without interrupting traffic flow. This scenario can be achieved by first reconfiguring the static BFD neighbor as detached (thus disassociating the neighbor from the static routes) and then deconfiguring the static BFD neighbor.

BFD Support for OSPFv3

BFD supports the dynamic routing protocol OSPF for IPv6 (OSPFv3). For information on how to configure OSPFv3, see the Configuring BFD Support for OSPFv3 section.

How to Configure Bidirectional Forwarding Detection for IPv6

Specifying a Static BFDv6 Neighbor

An IPv6 static BFDv6 neighbor is specified separately from an IPv6 static route. An IPv6 static BFDv6 neighbor must be fully configured with the interface and neighbor address and must be directly attached to the local router.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 route static bfd** [**vrf** *vrf-name*] *interface-type interface-number ipv6-address* [**unassociated**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 route static bfd** [**vrf** *vrf-name*] *interface-type interface-number ipv6-address* [**unassociated**] **Example:** `Device(config)# ipv6 route static bfd ethernet 0/0 2001:DB8::1`	Specifies static route IPv6 BFDv6 neighbors.

Associating an IPv6 Static Route with a BFDv6 Neighbor

IPv6 static routes are automatically associated with a static BFDv6 neighbor. A static neighbor is associated with a BFDv6 neighbor if the static next-hop explicitly matches the BFDv6 neighbor.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 route static bfd** [**vrf** *vrf-name*] *interface-type interface-number ipv6-address* [**unassociated**]
4. **ipv6 route** [**vrf** *vrf-name*] *ipv6-prefix/prefix-length prefix-length* {*ipv6-address* | *interface-type* [*interface-number ipv6-address*]} [**nexthop-vrf** [*vrf-name1* | **default**]] [*administrative-distance*] [*administrative-multicast-distance* | **unicast** | **multicast**] [*next-hop-address*] [**tag** *tag*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 route static bfd** [**vrf** *vrf-name*] *interface-type interface-number ipv6-address* [**unassociated**] **Example:** `Device(config)# ipv6 route static bfd ethernet 0/0 2001::1`	Specifies static route BFDv6 neighbors.
Step 4	**ipv6 route** [**vrf** *vrf-name*] *ipv6-prefix/prefix-length prefix-length* {*ipv6-address* | *interface-type* [*interface-number ipv6-address*]} [**nexthop-vrf** [*vrf-name1* | **default**]] [*administrative-distance*] [*administrative-multicast-distance* | **unicast** | **multicast**] [*next-hop-address*] [**tag** *tag*] **Example:** `Device(config)# ipv6 route 2001:DB8::/64 ethernet 0/0 2001::1`	Establishes static IPv6 routes.

Configuring BFD Support for OSPFv3

This section describes the procedures for configuring BFD support for OSPFv3, so that OSPFv3 is a registered protocol with BFD and will receive forwarding path detection failure messages from BFD. You can either configure BFD support for OSPFv3 globally on all interfaces or configure it selectively on one or more interfaces.

There are two methods for enabling BFD support for OSPFv3:

- You can enable BFD for all of the interfaces for which OSPFv3 is routing by using the **bfd all-interfaces** command in router configuration mode. You can disable BFD support on individual interfaces using the **ipv6 ospf bfd disable** command in interface configuration mode.
- You can enable BFD for a subset of the interfaces for which OSPFv3 is routing by using the **ipv6 ospf bfd** command in interface configuration mode.

Note OSPF will only initiate BFD sessions for OSPF neighbors that are in the FULL state.

Configuring Baseline BFD Session Parameters on the Interface

Repeat this task for each interface over which you want to run BFD sessions to BFD neighbors.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **bfd interval** *milliseconds* **min_rx** *milliseconds* **multiplier** *interval-multiplier*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface ethernet 0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**bfd interval** *milliseconds* **min_rx** *milliseconds* **multiplier** *interval-multiplier* **Example:** `Device(config-if)# bfd interval 50 min_rx 50 multiplier 5`	Enables BFD on the interface.

Configuring BFD Support for OSPFv3 for All Interfaces

OSPFv3 must be running on all participating routers. The baseline parameters for BFD sessions on the interfaces over which you want to run BFD sessions to BFD neighbors must be configured.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router ospf** *process-id* [**vrf** *vpn-name*]
4. **bfd all-interfaces**
5. **exit**
6. **show bfd neighbors** [**vrf** *vrf-name*] [**client** {**bgp** | **eigrp** | **isis** | **ospf** | **rsvp** | **te-frr**}] [*ip-address* | **ipv6** *ipv6-address*] [**details**]
7. **show ipv6 ospf** [*process-id*] [*area-id*] [**rate-limit**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router ospf** *process-id* [**vrf** *vpn-name*] **Example:** `Device(config)# ipv6 router ospf 2`	Configures an OSPFv3 routing process.
Step 4	**bfd all-interfaces** **Example:** `Device(config-router)# bfd all-interfaces`	Enables BFD for all interfaces participating in the routing process.

	Command or Action	Purpose
Step 5	**exit** **Example:** `Device(config-router)# exit`	Enter this command twice to go to privileged EXEC mode.
Step 6	**show bfd neighbors** [**vrf** *vrf-name*] [**client** {**bgp** | **eigrp** | **isis** | **ospf** | **rsvp** | **te-frr**}] [*ip-address* | **ipv6** *ipv6-address*] [**details**] **Example:** `Device# show bfd neighbors detail`	(Optional) Displays a line-by-line listing of existing BFD adjacencies.
Step 7	**show ipv6 ospf** [*process-id*] [*area-id*] [**rate-limit**] **Example:** `Device# show ipv6 ospf`	(Optional) Displays general information about OSPFv3 routing processes.

Configuring BFDv6 Support for OSPFv3 on One or More OSPFv3 Interfaces

OSPFv3 must be running on all participating routers. The baseline parameters for BFD sessions on the interfaces over which you want to run BFD sessions to BFD neighbors must be configured.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 ospf bfd** [**disable**]
5. **exit**
6. **show bfd neighbors** [**vrf** *vrf-name*] [**client** {**bgp** | **eigrp** | **isis** | **ospf** | **rsvp** | **te-frr**}] [*ip-address* | **ipv6** *ipv6-address*] [**details**]
7. **show ipv6 ospf** [*process-id*] [*area-id*] [**rate-limit**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

Command or Action	Purpose
Step 2 **configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3 **interface** *type number* **Example:** `Device(config)# interface ethernet 0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4 **ipv6 ospf bfd** [**disable**] **Example:** `Device(config-if)# ipv6 ospf bfd`	Enables BFD on a per-interface basis for one or more interfaces associated with the OSPFv3 routing process.
Step 5 **exit** **Example:** `Device(config-router)# exit`	Enter this command twice to go to privileged EXEC mode.
Step 6 **show bfd neighbors** [**vrf** *vrf-name*] [**client** {**bgp** | **eigrp** | **isis** | **ospf** | **rsvp** | **te-frr**}] [*ip-address* | **ipv6** *ipv6-address*] [**details**] **Example:** `Device# show bfd neighbors detail`	(Optional) Displays a line-by-line listing of existing BFD adjacencies.
Step 7 **show ipv6 ospf** [*process-id*] [*area-id*] [**rate-limit**] **Example:** `Device# show ipv6 ospf`	(Optional) Displays general information about OSPFv3 routing processes.

Retrieving BFDv6 Information for Monitoring and Troubleshooting

SUMMARY STEPS

1. **enable**
2. **monitor event ipv6 static** [**enable** | **disable**]
3. **show ipv6 static** [*ipv6-address* | *ipv6-prefix/prefix-length*] [**interface** *type number* | **recursive**] [**vrf** *vrf-name*] [**bfd**] [**detail**
4. **show ipv6 static** [*ipv6-address* | *ipv6-prefix/prefix-length*] [**interface** *type number* | **recursive**] [**vrf** *vrf-name*] [**bfd**] [**detail**]
5. **debug ipv6 static**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**monitor event ipv6 static** [**enable**	**disable**] **Example:** `Device# monitor event ipv6 static enable`	Enables the use of event trace to monitor the operation of the IPv6 static and IPv6 static BFDv6 neighbors.
Step 3	**show ipv6 static** [*ipv6-address*	*ipv6-prefix/prefix-length*] [**interface** *type number*	**recursive**] [**vrf** *vrf-name*] [**bfd**] [**detail** **Example:** `Device# show ipv6 static vrf vrf1 detail`	Displays the BFDv6 status for a static route associated with a static BFDv6 neighbor.
Step 4	**show ipv6 static** [*ipv6-address*	*ipv6-prefix/prefix-length*] [**interface** *type number*	**recursive**] [**vrf** *vrf-name*] [**bfd**] [**detail**] **Example:** `Device# show ipv6 static vrf vrf1 bfd`	Displays static BFDv6 neighbors and associated static routes.
Step 5	**debug ipv6 static** **Example:** `Device# debug ipv6 static`	Enables BFDv6 debugging.

Configuration Examples for Bidirectional Forwarding Detection for IPv6

Example: Specifying an IPv6 Static BFDv6 Neighbor

The following example shows how to specify a fully configured IPv6 static BFDv6 neighbor. The interface is Ethernet 0/0 and the neighbor address is 2001::1.

```
Device(config)# ipv6 route static bfd ethernet 0/0 2001::1
```

Example: Associating an IPv6 Static Route with a BFDv6 Neighbor

In this example, the IPv6 static route 2001:DB8::/32 is associated with the BFDv6 neighbor 2001::1 over the Ethernet 0/0 interface:

```
Device(config)# ipv6 route static bfd ethernet 0/0 2001::1
Device(config)# ipv6 route 2001:DB8::/32 ethernet 0/0 2001::1
```

Example: Displaying OSPF Interface Information about BFD

The following display shows that the OSPF interface is enabled for BFD:

```
Device# show ipv6 ospf interface

Serial10/0 is up, line protocol is up
  Link Local Address FE80::A8BB:CCFF:FE00:6500, Interface ID 42
  Area 1, Process ID 1, Instance ID 0, Router ID 10.0.0.1
  Network Type POINT_TO_POINT, Cost: 64
  Transmit Delay is 1 sec, State POINT_TO_POINT, BFD enabled
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    Hello due in 00:00:07
  Index 1/1/1, flood queue length 0
  Next 0x0(0)/0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 1
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 10.1.0.1
  Suppress hello for 0 neighbor(s)
```

Additional References

Related Documents

Related Topic	Document Title
OSPF for IPv6	"Implementing OSPF for IPv6," *Cisco IOS IPv6 Configuration Guide*
IPv6 static routes	"Implementing Static Routes for IPv6," *Cisco IOS IPv6 Configuration Guide*

Standards

Standard	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
draft-ietf-bfd-v4v6-1hop-07.txt	*BFD for IPv4 and IPv6 (Single Hop)*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing Bidirectional Forwarding Detection for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 11 Feature Information for Implementing Bidirectional Forwarding Detection for IPv6

Feature Name	Releases	Feature Information
OSPFv3 for BFD	12.2(33)SRE 15.0(1)S 15.0(1)SY 15.1(2)T	BFD supports the dynamic routing protocol OSPF for IPv6 (OSPFv3). The following commands were introduced or modified: **bfd**, **bfd all-interfaces**, **debug bfd**, **ipv6 router ospf**, **show bfd neighbors**, **show ipv6 ospf, show ipv6 ospf interface**.
BFD IPv6 Encapsulation Support	12.2(33)SRE 15.0(1)SY 15.1(2)T	BFDv6 encapsulations are described within a session information structure. These session information structures are defined by BFDv6 for the protocols supported. BFDv6 uses information from the session information structures to determine the correct encapsulation for BFDv6 packets on that session.

Feature Name	Releases	Feature Information
Static Route Support for BFD over IPv6	15.0(1)SY1 15.1(2)T	Using the BFDv6 protocol to reach the static route next hop ensures that an IPv6 static route is inserted only in the IPv6 Routing Information Base (RIB) when the next-hop neighbor is reachable. Using the BFDv6 protocol also can remove the IPv6 static route from the IPv6 RIB when the next hop becomes unreachable. The following commands were introduced or modified: **debug bfd**, **debug ipv6 static**, **ipv6 route**, **ipv6 route static bfd**, **monitor event ipv6 static**, **show ipv6 static**.

Implementing Multiprotocol BGP for IPv6

This module describes how to configure multiprotocol Border Gateway Protocol (BGP) for IPv6. BGP is an Exterior Gateway Protocol (EGP) used mainly to connect separate routing domains that contain independent routing policies (autonomous systems). Connecting to a service provider for access to the Internet is a common use for BGP. BGP can also be used within an autonomous system, and this variation is referred to as internal BGP (iBGP). Multiprotocol BGP is an enhanced BGP that carries routing information for multiple network layer protocol address families; for example, the IPv6 address family and IP multicast routes. All BGP commands and routing policy capabilities can be used with multiprotocol BGP.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Implementing Multiprotocol BGP for IPv6

Multiprotocol BGP Extensions for IPv6

Multiprotocol BGP is the supported exterior gateway protocol (EGP) for IPv6. Multiprotocol BGP extensions for IPv6 supports many of the same features and functionality as IPv4 BGP. IPv6 enhancements to multiprotocol BGP include support for an IPv6 address family and network layer reachability

information (NLRI) and next hop (the next router in the path to the destination) attributes that use IPv6 addresses.

- IPv6 Multiprotocol BGP Peering Using a Link-Local Address, page 148

IPv6 Multiprotocol BGP Peering Using a Link-Local Address

The IPv6 multiprotocol BGP can be configured between two IPv6 devices (peers) using link-local addresses. For this function to work, the interface for the neighbor must be identified by using the **neighbor update-source** command, and a route map must be configured to set an IPv6 global next hop.

Multiprotocol BGP for the IPv6 Multicast Address Family

The multiprotocol BGP for the IPv6 multicast address family feature provides multicast BGP extensions for IPv6 and supports the same features and functionality as IPv4 BGP. IPv6 enhancements to multicast BGP include support for an IPv6 multicast address family and network layer reachability information (NLRI) and next hop (the next router in the path to the destination) attributes that use IPv6 addresses.

Multicast BGP is an enhanced BGP that allows the deployment of interdomain IPv6 multicast. Multiprotocol BGP carries routing information for multiple network layer protocol address families; for example, IPv6 address family and for IPv6 multicast routes. The IPv6 multicast address family contains routes used for RPF lookup by the IPv6 PIM protocol, and multicast BGP IPv6 provides for interdomain transport of the same. Users must use multiprotocol BGP for IPv6 multicast when using IPv6 multicast with BGP because the unicast BGP learned routes will not be used for IPv6 multicast.

Multicast BGP functionality is provided through a separate address family context. A subsequent address family identifier (SAFI) provides information about the type of the network layer reachability information that is carried in the attribute. Multiprotocol BGP unicast uses SAFI 1 messages, and multiprotocol BGP multicast uses SAFI 2 messages. SAFI 1 messages indicate that the routes are usable only for IP unicast, not IP multicast. Because of this functionality, BGP routes in the IPv6 unicast RIB must be ignored in the IPv6 multicast RPF lookup.

A separate BGP routing table is maintained to configure incongruent policies and topologies (for example, IPv6 unicast and multicast) by using IPv6 multicast RPF lookup. Multicast RPF lookup is very similar to the IP unicast route lookup.

No MRIB is associated with the IPv6 multicast BGP table. However, IPv6 multicast BGP operates on the unicast IPv6 RIB when needed. Multicast BGP does not insert or update routes into the IPv6 unicast RIB.

- Nonstop Forwarding and Graceful Restart for MP-BGP IPv6 Address Family, page 148

Nonstop Forwarding and Graceful Restart for MP-BGP IPv6 Address Family

The graceful restart capability is supported for IPv6 BGP unicast, multicast, and VPNv6 address families, enabling Cisco nonstop forwarding (NSF) functionality for BGP IPv6. The BGP graceful restart capability allows the BGP routing table to be recovered from peers without keeping the TCP state.

NSF continues forwarding packets while routing protocols converge, therefore avoiding a route flap on switchover. Forwarding is maintained by synchronizing the FIB between the active and standby RP. On switchover, forwarding is maintained using the FIB. The RIB is not kept synchronized; therefore, the RIB is empty on switchover. The RIB is repopulated by the routing protocols and subsequently informs FIB about RIB convergence by using the NSF_RIB_CONVERGED registry call. The FIB tables are updated from the RIB, removing any stale entries. The RIB starts a failsafe timer during RP switchover, in case the routing protocols fail to notify the RIB of convergence.

The Cisco BGP address family identifier (AFI) model is designed to be modular and scalable, and to support multiple AFI and subsequent address family identifier (SAFI) configurations.

How to Implement Multiprotocol BGP for IPv6

When configuring multiprotocol BGP extensions for IPv6, you must create the BGP routing process, configure peering relationships, and customize BGP for your particular network.

Configuring an IPv6 BGP Routing Process and BGP Router ID

Perform this task to configure an IPv6 BGP routing process and an optional BGP router ID for a BGP-speaking device.

BGP uses a router ID to identify BGP-speaking peers. The BGP router ID is 32-bit value that is often represented by an IPv4 address. By default, the router ID is set to the IPv4 address of a loopback interface on the device. If no loopback interface is configured on the device, then the software chooses the highest IPv4 address configured to a physical interface on the device to represent the BGP router ID.

When configuring BGP on a device that is enabled only for IPv6 (the device does not have an IPv4 address), you must manually configure the BGP router ID for the device. The BGP router ID, which is represented as a 32-bit value using an IPv4 address syntax, must be unique to the BGP peers of the device.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **no bgp default ipv4-unicast**
5. **bgp router-id** *ip-address*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Configures a BGP routing process, and enters router configuration mode for the specified routing process.
Step 4	**no bgp default ipv4-unicast** **Example:** `Device(config-router)# no bgp default ipv4-unicast`	Disables the IPv4 unicast address family for the BGP routing process specified in the previous step. **Note** Routing information for the IPv4 unicast address family is advertised by default for each BGP routing session configured with the **neighbor remote-as** command unless you configure the **no bgp default ipv4-unicast** command before configuring the **neighbor remote-as** command.
Step 5	**bgp router-id** *ip-address* **Example:** `Device(config-router)# bgp router-id 192.168.99.70`	(Optional) Configures a fixed 32-bit router ID as the identifier of the local device running BGP. **Note** Configuring a router ID using the **bgp router-id** command resets all active BGP peering sessions.

Configuring IPv6 Multiprotocol BGP Between Two Peers

By default, neighbors that are defined using the **neighbor remote-as** command in router configuration mode exchange only IPv4 unicast address prefixes. To exchange other address prefix types, such as IPv6 prefixes, neighbors must also be activated using the **neighbor activate** command in address family configuration mode for the other prefix types, as shown for IPv6 prefixes.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number* ...]
5. **address-family ipv6** [**unicast** | **multicast**]
6. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* **%**} **activate**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number* ...] **Example:** `Device(config-router)# neighbor 2001:DB8:0:CC00::1 remote-as 64600`	Adds the IPv6 address of the neighbor in the specified autonomous system to the IPv6 multiprotocol BGP neighbor table of the local device.

	Command or Action	Purpose
Step 5	**address-family ipv6** [**unicast** | **multicast**] **Example:** `Device(config-router)# address-family ipv6`	Specifies the IPv6 address family and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the device is placed in configuration mode for the IPv6 unicast address family if a keyword is not specified with the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.
Step 6	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* **%**} **activate** **Example:** `Device(config-router-af)# neighbor 2001:DB8:0:CC00::1 activate`	Enables the neighbor to exchange prefixes for the IPv6 address family with the local device.

Configuring an IPv6 Multiprotocol BGP Peer Using a Link-Local Address

Configuring IPv6 multiprotocol BGP between two IPv6 devices (peers) using link-local addresses requires that the interface for the neighbor be identified by using the **neighbor update-source** command and that a route map be configured to set an IPv6 global next hop.

Note

- By default, neighbors that are defined using the **neighbor remote-as** command in router configuration mode exchange only IPv4 unicast address prefixes. To exchange other address prefix types, such as IPv6 prefixes, neighbors must also be activated using the **neighbor activate** command in address family configuration mode for the other prefix types, as shown for IPv6 prefixes.
- By default, route maps that are applied in router configuration mode using the **neighbor route-map** command are applied to only IPv4 unicast address prefixes. Route maps for other address families must be applied in address family configuration mode using the **neighbor route-map** command, as shown for the IPv6 address family. The route maps are applied either as the inbound or outbound routing policy for neighbors under the specified address family. Configuring separate route maps under each address family type simplifies managing complicated or different policies for each address family.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number* ...]
5. **neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **update-source** *interface-type interface-number*
6. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**]
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* **%**} **activate**
8. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*[**%**]} **route-map** *map-name* {**in** | **out**}
9. **exit**
10. **exit**
11. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]
12. **match ipv6 address** {**prefix-list** *prefix-list-name* | *access-list-name*}
13. **set ipv6 next-hop** *ipv6-address* [*link-local-address*] [**peer-address**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.

	Command or Action	Purpose
Step 4	**neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number ...*] **Example:** `Device(config-router)# neighbor FE80::1234:BFF:FE0E:A471% remote-as 64600`	Adds the link-local IPv6 address of the neighbor in the specified remote autonomous system to the IPv6 multiprotocol BGP neighbor table of the local device. • The optional **%** keyword is the IPv6 link-local address identifier. This keyword needs to be added whenever a link-local IPv6 address is used outside the context of its interface.
Step 5	**neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor FE80::1234:BFF:FE0E:A471% update-source fastethernet0`	Specifies the link-local address over which the peering is to occur. • The optional **%** keyword is the IPv6 link-local address identifier. This keyword needs to be added whenever a link-local IPv6 address is used outside the context of its interface. • If there are multiple connections to the neighbor and you do not specify the neighbor interface by using the *interface-type* and *interface-number* arguments in the **neighbor update-source** command, a TCP connection cannot be established with the neighbor using link-local addresses.
Step 6	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**] **Example:** `Device(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the device is placed in configuration mode for the IPv6 unicast address family if the **unicast** keyword is not specified with the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.
Step 7	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* **%**} **activate** **Example:** `Device(config-router-af)# neighbor FE80::1234:BFF:FE0E:A471% activate`	Enables the neighbor to exchange prefixes for the IPv6 address family with the local device using the specified link-local addresses. • The optional **%** keyword is the IPv6 link-local address identifier. This keyword needs to be added whenever a link-local IPv6 address is used outside the context of its interface.
Step 8	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*[**%**]} **route-map** *map-name* {**in** | **out**} **Example:** `Device(config-router-af)# neighbor FE80::1234:BFF:FE0E:A471% route-map nh6 out`	Applies a route map to incoming or outgoing routes. • The optional **%** keyword is the IPv6 link-local address identifier. This keyword needs to be added whenever a link-local IPv6 address is used outside the context of its interface.

	Command or Action	Purpose
Step 9	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode, and returns the device to router configuration mode.
Step 10	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode, and returns the device to global configuration mode.
Step 11	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map nh6 permit 10`	Defines a route map and enters route-map configuration mode.
Step 12	**match ipv6 address** {**prefix-list** *prefix-list-name* | *access-list-name*} **Example:** `Device(config-route-map)# match ipv6 address prefix-list cisco`	Distributes any routes that have a destination IPv6 network number address permitted by a prefix list, or performs policy routing on packets.
Step 13	**set ipv6 next-hop** *ipv6-address* [*link-local-address*] [**peer-address**] **Example:** `Device(config-route-map)# set ipv6 next-hop 2001:DB8::1`	Overrides the next hop advertised to the peer for IPv6 packets that pass a match clause of a route map for policy routing. • The *ipv6-address* argument specifies the IPv6 global address of the next hop. It need not be an adjacent device. • The *link-local-address* argument specifies the IPv6 link-local address of the next hop. It must be an adjacent device. If you do not specify this optional argument, the link-local address of the interface specified with the *interface-type* argument (in the **neighbor update-source** command in Step 5) is included as the next-hop in the BGP updates. Therefore, only one route map that sets the global IPv6 next-hop address in BGP updates is required for multiple BGP peers that use link-local addresses. • The route map sets the IPv6 next-hop addresses (global and link-local) in BGP updates. If the route map is not configured, the next-hop address in the BGP updates defaults to the unspecified IPv6 address (::), which is rejected by the peer.

- Troubleshooting Tips, page 156

Troubleshooting Tips

If peering is not established by this task, it may be because of a missing route map **set ipv6 next-hop** command. Use the **debug bgp ipv6 update** command to display debugging information on the updates to help determine the state of the peering.

Configuring an IPv6 Multiprotocol BGP Peer Group

- By default, neighbors that are defined using the **neighbor remote-as** command in router configuration mode exchange only IPv4 unicast address prefixes. To exchange other address prefix types, such as IPv6 prefixes, neighbors must also be activated using the **neighbor activate** command in address family configuration mode for the other prefix types, as shown for IPv6 prefixes.
- By default, peer groups that are defined in router configuration mode using the **neighbor peer-group**command exchange only IPv4 unicast address prefixes. To exchange other address prefix types, such as IPv6 prefixes, you must activate peer groups using the **neighbor activate** command in address family configuration mode for the other prefix types, as shown for IPv6 prefixes.
- Members of a peer group automatically inherit the address prefix configuration of the peer group.
- IPv4 active neighbors cannot exist in the same peer group as active IPv6 neighbors. Create separate peer groups for IPv4 peers and IPv6 peers.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **neighbor** *peer-group-name* **peer-group**
5. **neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number* ...]
6. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* **%**} **activate**
8. **neighbor** *ip-address* | *ipv6-address*} **send-label**
9. **neighbor** {*ip-address* | *ipv6-address*} **peer-group** *peer-group-name*
10. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Router(config)# router bgp 65000`	Enters router configuration mode for the specified BGP routing process.
Step 4	**neighbor** *peer-group-name* **peer-group** **Example:** `Router(config-router)# neighbor group1 peer-group`	Creates a multiprotocol BGP peer group.
Step 5	**neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number* ...] **Example:** `Router(config-router)# neighbor 2001:DB8:0:CC00::1 remote-as 64600`	Adds the IPv6 address of the neighbor in the specified autonomous system to the IPv6 multiprotocol BGP neighbor table of the local router.
Step 6	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6** **Example:** `Router(config-router)# address-family ipv6 unicast`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the router is placed in configuration mode for the IPv6 unicast address family if the **unicast** keyword is not specified with the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.
Step 7	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* **%**} **activate** **Example:** `Router(config-router-af)# neighbor 2001:DB8:0:CC00::1 activate`	Enables the neighbor to exchange prefixes for the specified family type with the neighbor and the local router. • To avoid extra configuration steps for each neighbor, use the **neighbor activate** command with the *peer-group-name* argument as an alternative in this step.

	Command or Action	Purpose
Step 8	**neighbor** *ip-address* | *ipv6-address*} **send-label** **Example:** `Router(config-router-af)# neighbor 192.168.99.70 send-label`	Advertises the capability of the router to send MPLS labels with BGP routes. • In IPv6 address family configuration mode, this command enables binding and advertisement of aggregate labels when advertising IPv6 prefixes in BGP.
Step 9	**neighbor** {*ip-address* | *ipv6-address*} **peer-group** *peer-group-name* **Example:** `Router(config-router-af)# neighbor 2001:DB8:0:CC00::1 peer-group group1`	Assigns the IPv6 address of a BGP neighbor to a peer group.
Step 10	**exit** **Example:** `Router(config-router-af)# exit`	Exits address family configuration mode, and returns the router to router configuration mode. • Repeat this step to exit router configuration mode and return the router to global configuration mode.

Advertising Routes into IPv6 Multiprotocol BGP

By default, networks that are defined in router configuration mode using the **network** command are injected into the IPv4 unicast database. To inject a network into another database, such as the IPv6 BGP database, you must define the network using the **network** command in address family configuration mode for the other database, as shown for the IPv6 BGP database.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**]
5. **network** {*network-number* [**mask** *network-mask*] | *nsap-prefix*} [**route-map** *map-tag*]
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified BGP routing process.
Step 4	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**] **Example:** `Device(config-router)# address-family ipv6 unicast`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the router is placed in configuration mode for the IPv6 unicast address family if a keyword is not specified with the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.
Step 5	**network** {*network-number* [**mask** *network-mask*] | *nsap-prefix*} [**route-map** *map-tag*] **Example:** `Device(config-router-af)# network 2001:DB8::/24`	Advertises (injects) the specified prefix into the IPv6 BGP database. (The routes must first be found in the IPv6 unicast routing table.) • Specifically, the prefix is injected into the database for the address family specified in the previous step. • Routes are tagged from the specified prefix as "local origin." • The *ipv6-prefix*argument in the **network** command must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons. • The *prefix-length* argument is a decimal value that indicates how many of the high-order contiguous bits of the address comprise the prefix (the network portion of the address). A slash mark must precede the decimal value.
Step 6	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode, and returns the router to router configuration mode. • Repeat this step to exit router configuration mode and return the router to global configuration mode.

Configuring a Route Map for IPv6 Multiprotocol BGP Prefixes

- By default, neighbors that are defined using the **neighbor remote-as** command in router configuration mode exchange only IPv4 unicast address prefixes. To exchange other address prefix types, such as IPv6 prefixes, neighbors must also be activated using the **neighbor activate** command in address family configuration mode for the other prefix types, as shown for IPv6 prefixes.
- By default, route maps that are applied in router configuration mode using the **neighbor route-map** command are applied to only IPv4 unicast address prefixes. Route maps for other address families must be applied in address family configuration mode using the **neighbor route-map** command, as shown for the IPv6 address family. The route maps are applied either as the inbound or outbound routing policy for neighbors under the specified address family. Configuring separate route maps under each address family type simplifies managing complicated or different policies for each address family.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number ...*]
5. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**]
6. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* **%**} **activate**
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* [**%**]} **route-map** *map-name* {**in** | **out**}
8. **exit**
9. **exit**
10. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]
11. **match ipv6 address** {**prefix-list** *prefix-list-name* | *access-list-name*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number* ...] **Example:** `Device(config-router)# neighbor 2001:DB8:0:cc00::1 remote-as 64600`	Adds the link-local IPv6 address of the neighbor in the specified remote autonomous system to the IPv6 multiprotocol BGP neighbor table of the local device.
Step 5	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**] **Example:** `Device(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the device is placed in configuration mode for the IPv6 unicast address family if the **unicast** keyword is not specified with the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.
Step 6	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* **%**} **activate** **Example:** `Device(config-router-af)# neighbor 2001:DB8:0:cc00::1 activate`	Enables the neighbor to exchange prefixes for the IPv6 address family with the local device using the specified link-local addresses.
Step 7	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* [**%**]} **route-map** *map-name* {**in** | **out**} **Example:** `Device(config-router-af)# neighbor 2001:DB8:0:cc00::1 route-map rtp in`	Applies a route map to incoming or outgoing routes. • Changes to the route map will not take effect for existing peers until the peering is reset or a soft reset is performed. Using the **clear bgp ipv6** command with the **soft** and **in** keywords will perform a soft reset.
Step 8	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode, and returns the device to router configuration mode.

	Command or Action	Purpose
Step 9	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode, and returns the device to global configuration mode.
Step 10	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map rtp permit 10`	Defines a route map and enters route-map configuration mode. • Follow this step with a **match** command.
Step 11	**match ipv6 address** {**prefix-list** *prefix-list-name* | *access-list-name*} **Example:** `Device(config-route-map)# match ipv6 address prefix-list cisco`	Distributes any routes that have a destination IPv6 network number address permitted by a prefix list, or performs policy routing on packets.

Redistributing Prefixes into IPv6 Multiprotocol BGP

Redistribution is the process of redistributing, or injecting, prefixes from one routing protocol into another routing protocol. This task explains how to inject prefixes from a routing protocol into IPv6 multiprotocol BGP. Specifically, prefixes that are redistributed into IPv6 multiprotocol BGP using the **redistribute** router configuration command are injected into the IPv6 unicast database.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**]
5. **redistribute bgp** [*process-id*] [**metric** *metric-value*] [**route-map** *map-name*] [*source-protocol-options*]
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified BGP routing process.
Step 4	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**] **Example:** `Device(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the router is placed in configuration mode for the IPv6 unicast address family if a keyword is not specified with the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.
Step 5	**redistribute bgp** [*process-id*] [**metric** *metric-value*] [**route-map** *map-name*] [*source-protocol-options*] **Example:** `Device(config-router-af)# redistribute bgp 64500 metric 5 metric-type external`	Redistributes IPv6 routes from one routing domain into another routing domain.
Step 6	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode, and returns the router to router configuration mode. • Repeat this step to exit router configuration mode and return the router to global configuration mode.

Advertising IPv4 Routes Between IPv6 BGP Peers

If an IPv6 network is connecting two separate IPv4 networks, it is possible to use IPv6 to advertise the IPv4 routes. Configure the peering using the IPv6 addresses within the IPv4 address family. Set the next hop with a static route or with an inbound route map because the advertised next hop will usually be unreachable. Advertising IPv6 routes between two IPv4 peers is also possible using the same model.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **neighbor** *peer-group-name* **peer-group**
5. **neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number* ...]
6. **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*
7. **neighbor** *ipv6-address* **peer-group** *peer-group-name*
8. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* [**%**]} **route-map** *map-name* {**in** | **out**}
9. **exit**
10. **exit**
11. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]
12. **set ip next-hop** *ip-address* [... *ip-address*] [**peer-address**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.

	Command or Action	Purpose
Step 4	**neighbor** *peer-group-name* **peer-group** **Example:** `Device(config-router)# neighbor 6peers peer-group`	Creates a multiprotocol BGP peer group.
Step 5	**neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number* ...] **Example:** `Device(config-router)# neighbor 6peers remote-as 65002`	Adds the IPv6 address of the neighbor in the specified autonomous system to the IPv6 multiprotocol BGP neighbor table of the local router.
Step 6	**address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name* **Example:** `Device(config-router)# address-family ipv4`	Enters address family configuration mode to configure a routing session using standard IPv4 address prefixes.
Step 7	**neighbor** *ipv6-address* **peer-group** *peer-group-name* **Example:** `Device(config-router-af)# neighbor 2001:DB8:1234::2 peer-group 6peers`	Assigns the IPv6 address of a BGP neighbor to a peer group.
Step 8	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* [**%**]} **route-map** *map-name* {**in** | **out**} **Example:** `Device(config-router-af)# neighbor 6peers route-map rmap out`	Applies a route map to incoming or outgoing routes. • Changes to the route map will not take effect for existing peers until the peering is reset or a soft reset is performed. Using the **clear bgp ipv6** command with the **soft** and **in** keywords will perform a soft reset.
Step 9	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode, and returns the device to router configuration mode.

	Command or Action	Purpose
Step 10	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode, and returns the device to global configuration mode.
Step 11	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map rmap permit 10`	Defines a route map and enters route-map configuration mode.
Step 12	**set ip next-hop** *ip-address* [*... ip-address*] [**peer-address**] **Example:** `Device(config-route-map)# set ip next-hop 10.21.8.10`	Overrides the next hop advertised to the peer for IPv4 packets.

Assigning BGP Administrative Distance for Multicast BGP Routes

Perform this task to specify an administrative distance for multicast BGP routes to be used in RPF lookups for comparison with unicast routes.

Caution Changing the administrative distance of BGP internal routes is not recommended. One problem that can occur is the accumulation of routing table inconsistencies, which can break routing.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**]
5. **distance bgp** *external-distance internal-distance local-distance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Router(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.
Step 4	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**] **Example:** `Router(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the router is placed in configuration mode for the IPv6 unicast address family if the **unicast** keyword is not specified with the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.
Step 5	**distance bgp** *external-distance internal-distance local-distance* **Example:** `Router(config-router-af)# distance bgp 10 50 100`	Configures the administrative distance for BGP routes.

Generating IPv6 Multicast BGP Updates

Perform this task to generate IPv6 multicast BGP updates that correspond to unicast IPv6 updates received from a peer.

The MBGP translate-update feature generally is used in an MBGP-capable router that peers with a customer site that has only a BGP-capable router; the customer site has not or cannot upgrade its router to an MBGP-capable image. Because the customer site cannot originate MBGP advertisements, the router with which it peers will translate the BGP prefixes into MBGP prefixes, which are used for multicast-source Reverse Path Forwarding (RPF) lookup.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**
5. **neighbor** *ipv6-address* **translate-update ipv6 multicast** [**unicast**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Router(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.
Step 4	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6** **Example:** `Router(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the router is placed in configuration mode for the IPv6 unicast address family if the **unicast** keyword is not specified with the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.
Step 5	**neighbor** *ipv6-address* **translate-update ipv6 multicast** [**unicast** **Example:** `Router(config-router-af)# neighbor 7000::2 translate-update ipv6 multicast`	Generates multiprotocol IPv6 BGP updates that correspond to unicast IPv6 updates received from a peer.

Configuring the IPv6 BGP Graceful Restart Capability

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**]
5. **bgp graceful-restart** [**restart-time** *seconds* | **stalepath-time** *seconds*] [**all**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.
Step 4	**address-family ipv6** [**vrf** *vrf-name*] [**unicast**	**multicast**	**vpnv6**] **Example:** `Device(config-router)# address-family ipv6`	Specifies the IPv6 address family.
Step 5	**bgp graceful-restart** [**restart-time** *seconds*	**stalepath-time** *seconds*] [**all**] **Example:** `Device(config-router-af)# bgp graceful-restart`	Enables the BGP graceful restart capability.

Resetting IPv6 BGP Sessions

SUMMARY STEPS

1. **enable**
2. **clear bgp ipv6** {**unicast** | **multicast**} {***** | *autonomous-system-number* | *ip-address* | *ipv6-address* | **peer-group** *peer-group-name*} [**soft**] [**in** | **out**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear bgp ipv6** {**unicast**	**multicast**} {*****	*autonomous-system-number*	*ip-address*	*ipv6-address*	**peer-group** *peer-group-name*} [**soft**] [**in**	**out**] **Example:** `Device# clear bgp ipv6 unicast peer-group marketing soft out`	Resets IPv6 BGP sessions.

Clearing External BGP Peers

SUMMARY STEPS

1. **enable**
2. **clear bgp ipv6** {**unicast** | **multicast**} **external** [**soft**] [**in** | **out**]
3. **clear bgp ipv6** {**unicast** | **multicast**} **peer-group** *name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**clear bgp ipv6** {**unicast** | **multicast**} **external** [**soft**] [**in** | **out**] **Example:** `Device# clear bgp ipv6 unicast external soft in`	Clears external IPv6 BGP peers.
Step 3	**clear bgp ipv6** {**unicast** | **multicast**} **peer-group** *name* **Example:** `Device# clear bgp ipv6 unicast peer-group marketing`	Clears all members of an IPv6 BGP peer group.

Clearing IPv6 BGP Route Dampening Information

SUMMARY STEPS

1. **enable**
2. **clear bgp ipv6** {**unicast** | **multicast**} **dampening** [*ipv6-prefix prefix-length*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear bgp ipv6** {**unicast** | **multicast**} **dampening** [*ipv6-prefix prefix-length*] **Example:** `Device# clear bgp ipv6 unicast dampening 2001:DB8::/64`	Clears IPv6 BGP route dampening information and unsuppresses the suppressed routes.

Clearing IPv6 BGP Flap Statistics

SUMMARY STEPS

1. **enable**
2. **clear bgp ipv6** {**unicast** | **multicast**} **flap-statistics** [*ipv6-prefix*/*prefix-length* | **regexp** *regexp* | **filter-list** *list*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear bgp ipv6** {**unicast** | **multicast**} **flap-statistics** [*ipv6-prefix*/*prefix-length* | **regexp** *regexp* | **filter-list** *list*] **Example:** `Device# clear bgp ipv6 unicast flap-statistics filter-list 3`	Clears IPv6 BGP flap statistics.

Verifying IPv6 Multiprotocol BGP Configuration and Operation

SUMMARY STEPS

1. **show bgp ipv6 unicast** | **multicast**} [*ipv6-prefix*/*prefix-length*] [**longer-prefixes**] [**labels**]
2. **show bgp ipv6** {**unicast** | **multicast**} **summary**
3. **show bgp ipv6** {**unicast** | **multicast**} **dampening dampened-paths**
4. **enable**
5. **debug bgp ipv6** {**unicast** | **multicast**} **dampening**[**prefix-list** *prefix-list-name*]
6. **debug bgp ipv6 unicast** | **multicast**} **updates**[*ipv6-address*] [**prefix-list** *prefix-list-name*] [**in**| **out**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show bgp ipv6 unicast** | **multicast**} [*ipv6-prefix*/*prefix-length*] [**longer-prefixes**] [**labels**] **Example:** `Router> show bgp ipv6 unicast`	(Optional) Displays entries in the IPv6 BGP routing table.
Step 2	**show bgp ipv6** {**unicast** | **multicast**} **summary** **Example:** `Router> show bgp ipv6 unicast summary`	(Optional) Displays the status of all IPv6 BGP connections.

	Command or Action	Purpose
Step 3	**show bgp ipv6** {**unicast** | **multicast**} **dampening dampened-paths** **Example:** `Router> show bgp ipv6 unicast dampening dampened-paths`	(Optional) Displays IPv6 BGP dampened routes.
Step 4	**enable** **Example:** `Router> enable`	Enables higher privilege levels, such as privileged EXEC mode. • Enter your password if prompted.
Step 5	**debug bgp ipv6** {**unicast** | **multicast**} **dampening**[**prefix-list** *prefix-list-name*] **Example:** `Router# debug bgp ipv6 unicast dampening`	(Optional) Displays debugging messages for IPv6 BGP dampening packets. • If no prefix list is specified, debugging messages for all IPv6 BGP dampening packets are displayed.
Step 6	**debug bgp ipv6 unicast** | **multicast**} **updates**[*ipv6-address*] [**prefix-list** *prefix-list-name*] [**in**| **out**] **Example:** `Router# debug bgp ipv6 unicast updates`	(Optional) Displays debugging messages for IPv6 BGP update packets. • If an *ipv6-address* argument is specified, debugging messages for IPv6 BGP updates to the specified neighbor are displayed. • Use the **in** keyword to display debugging messages for inbound updates only. • Use the **out** keyword to display debugging messages for outbound updates only.

Configuration Examples for Multiprotocol BGP for IPv6

Example: Configuring a BGP Process, BGP Router ID, and IPv6 Multiprotocol BGP Peer

The following example enables IPv6 globally, configures a BGP process, and establishes a BGP router ID. Also, the IPv6 multiprotocol BGP peer 2001:DB8:0:CC00:: is configured and activated.

```
ipv6 unicast-routing
!
router bgp 65000
no bgp default ipv4-unicast
bgp router-id 192.168.99.70
neighbor 2001:DB8:0:CC00::1 remote-as 64600
address-family ipv6 unicast
  neighbor 2001:DB8:0:CC00::1 activate
```

Example: Configuring an IPv6 Multiprotocol BGP Peer Using a Link-Local Address

The following example configures the IPv6 multiprotocol BGP peer FE80::1234:BFF:FE0E:A471 over Fast Ethernet interface 0 and sets the route map named nh6 to include the IPv6 next-hop global address of Fast Ethernet interface 0 in BGP updates. The IPv6 next-hop link-local address can be set by the nh6 route map (not shown in the following example) or from the interface specified by the **neighbor update-source** command (as shown in the following example).

```
router bgp 65000
 neighbor FE80::1234:BFF:FE0E:A471 remote-as 64600
 neighbor FE80::1234:BFF:FE0E:A471 update-source fastethernet 0
address-family ipv6
 neighbor FE80::1234:BFF:FE0E:A471 activate
 neighbor FE80::1234:BFF:FE0E:A471 route-map nh6 out
route-map nh6 permit 10
 match ipv6 address prefix-list cisco
 set ipv6 next-hop 2001:DB8:526::1
ipv6 prefix-list cisco permit 2001:DB8:2F22::/48 le 128
ipv6 prefix-list cisco deny ::/0
```

Note If you specify only the global IPv6 next-hop address (the *ipv6-address* argument) with the **set ipv6 next-hop** command after specifying the neighbor interface (the *interface-type* argument) with the **neighbor update-source** command, the link-local address of the interface specified with the *interface-type* argument is included as the next hop in the BGP updates. Therefore, only one route map that sets the global IPv6 next-hop address in BGP updates is required for multiple BGP peers that use link-local addresses.

Example: Configuring an IPv6 Multiprotocol BGP Peer Group

The following example configures the IPv6 multiprotocol BGP peer group named group1:

```
router bgp 65000
no bgp default ipv4-unicast
neighbor group1 peer-group
neighbor 2001:DB8:0:CC00::1 remote-as 64600
address-family ipv6 unicast
 neighbor group1 activate
 neighbor 2001:DB8:0:CC00::1 peer-group group1
```

Example: Advertising Routes into IPv6 Multiprotocol BGP

The following example injects the IPv6 network 2001:DB8::/24 into the IPv6 unicast database of the local router. (BGP checks that a route for the network exists in the IPv6 unicast database of the local router before advertising the network.)

```
router bgp 65000
 no bgp default ipv4-unicast
address-family ipv6 unicast
  network 2001:DB8::/24
```

Example: Configuring a Route Map for IPv6 Multiprotocol BGP Prefixes

The following example configures the route map named rtp to permit IPv6 unicast routes from network 2001:DB8::/24 if they match the prefix list named cisco:

```
router bgp 64900
no bgp default ipv4-unicast
neighbor 2001:DB8:0:CC00::1 remote-as 64700
address-family ipv6 unicast
 neighbor 2001:DB8:0:CC00::1 activate
 neighbor 2001:DB8:0:CC00::1 route-map rtp in
ipv6 prefix-list cisco seq 10 permit 2001:DB8::/24
route-map rtp permit 10
 match ipv6 address prefix-list cisco
```

Example: Redistributing Prefixes into IPv6 Multiprotocol BGP

The following example redistributes RIP routes into the IPv6 unicast database of the local router:

```
router bgp 64900
no bgp default ipv4-unicast
address-family ipv6 unicast
 redistribute rip
```

Example: Advertising IPv4 Routes Between IPv6 Peers

The following example advertises IPv4 routes between IPv6 peers when the IPv6 network is connecting two separate IPv4 networks. Peering is configured using IPv6 addresses in the IPv4 address family configuration mode. The inbound route map named rmap sets the next hop because the advertised next hop is likely to be unreachable.

```
router bgp 65000
!
 neighbor 6peers peer-group
 neighbor 2001:DB8:1234::2 remote-as 65002
 address-family ipv4
 neighbor 6peers activate
 neighbor 6peers soft-reconfiguration inbound
 neighbor 2001:DB8:1234::2 peer-group 6peers
 neighbor 2001:DB8:1234::2 route-map rmap in
!
route-map rmap permit 10
 set ip next-hop 10.21.8.10
```

Additional References

Related Documents

Related Topic	Document Title
IPv4 BGP configuration tasks	*Cisco IOS IP Routing Protocols Configuration Guide*
Multiprotocol BGP configuration tasks	*Cisco IOS IP Routing Protocols Configuration Guide*
BGP and multiprotocol BGP commands: complete command syntax, command mode, defaults, usage guidelines, and examples	"BGP Commands," *Cisco IOS IP Routing Protocols Command Reference*
Cisco nonstop forwarding	"Cisco Nonstop Forwarding," *Cisco IOS High Availability Configuration Guide*
IPv6 supported feature list	"Start Here: Cisco IOS Software Release Specifics for IPv6 Features," *Cisco IOS IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIBs	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 2545	*Use of BGP-4 Multiprotocol Extensions for IPv6 Inter-Domain Routing*
RFC 2858	*Multiprotocol Extensions for BGP-4*
RFC 4007	*IPv6 Scoped Address Architecture*
RFC 4364	BGP MPLS/IP Virtual Private Networks (VPNs)
RFC 4382	MPLS/BGP Layer 3 Virtual Private Network (VPN) Management Information Base
RFC 4659	BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN
RFC 4724	*Graceful Restart Mechanism for BGP*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing Multiprotocol BGPfor IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 12 Feature Information for Implementing Multiprotocol BGP for IPv6

Feature Name	Releases	Feature Information
6PE Multipath	12.2(25)S 12.2(28)SB 12.2(33)SRA 12.2(33)SXI1 12.4(6)T	The 6PE multipath feature uses multiprotocol internal BGP (MP-iBGP) to distribute IPv6 routes over the MPLS IPv4 core network and to attach an MPLS label to each route.
Advertising Routes into IPv6 Multiprotocol BGP	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T	Users advertise (inject) a prefix into IPv6 multiprotocol BGP.
Configuring Route Maps for IPv6 Multiprotocol BGP Prefixes	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T	Users can configure route maps for IPv6 multiprotocol BGP prefixes.
IPv6--NSF and Graceful Restart for MP-BGP IPv6 Address Family	12.2(33)SRE 12.2(33)XNE 15.0(1)SY	The graceful restart capability is supported for IPv6 BGP unicast, multicast, and VPNv6 address families, enabling Cisco nonstop forwarding (NSF) functionality for BGP IPv6. The BGP graceful restart capability allows the BGP routing table to be recovered from peers without keeping the TCP state.
IPv6 Multicast Address Family Support for Multiprotocol BGP	12.0(26)S 12.2(25)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(33)SXH 12.3(4)T 12.4 12.4(2)T	The multiprotocol BGP for the IPv6 multicast address family feature provides multicast BGP extensions for IPv6 and supports the same features and functionality as IPv4 BGP.
IPv6 Routing--Multiprotocol BGP Extensions for IPv6	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(17a)SX1 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	Multiprotocol BGP extensions for IPv6 supports the same features and functionality as IPv4 BGP.
IPv6 Routing--Multiprotocol BGP Link-Local Address Peering	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(17a)SX112.2(4)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	IPv6 supports multiprotocol BGP link-local address peering.
Redistributing Prefixes into IPv6 Multiprotocol BGP	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T	Users can redistribute (inject) prefixes from another routing protocol into IPv6 multiprotocol BGP.
VRF Lite Support for IPv6	12.2(58)SE	This feature is supported.

Implementing DHCP for IPv6

This module describes how to configure Dynamic Host Configuration Protocol (DHCP) for IPv6.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Implementing DHCP for IPv6

- Cisco IOS Release 12.0S provides IPv6 support on Gigabit Switch Routers (GSRs) and Cisco 10720 Internet routers only.
- The DHCPv6 Remote-ID for Ethernet Interfaces feature works only for Ethernet interfaces in Cisco IOS Release 12.2(33)SRC.
- The DHCPv6 implementation in Cisco IOS Release 12.3(4)T, Cisco IOS Release 12.0(32)S, and Cisco IOS 12.2(33)SRC supports only stateless address assignment.

Information About Implementing DHCP for IPv6

DHCPv6 Prefix Delegation

The IPv6 Access Services—DHCPv6 Prefix Delegation feature can be used to manage link, subnet, and site addressing changes. Dynamic Host Configuration Protocol for IPv6 (DHCPv6) can be used in environments to deliver stateful and stateless information. The definitions are given below:

- Stateful—Address assignment is centrally managed and clients must obtain configuration information that is not available through protocols such as address autoconfiguration and neighbor discovery.
- Stateless—Stateless configuration parameters do not require a server to maintain any dynamic state for individual clients, such as Domain Name System (DNS) server addresses and domain search list options.

Extensions to DHCPv6 also enable prefix delegation, through which an ISP can automate the process of assigning prefixes to a customer for use within the customer's network. Prefix delegation occurs between a provider edge (PE) device and customer premises equipment (CPE) using the DHCPv6 prefix delegation option. Once the ISP has delegated prefixes to a customer, the customer may further subnet and assign prefixes to the links in the customer's network.

Configuring Nodes Without Prefix Delegation

Stateless DHCPv6 allows DHCPv6 to be used for configuring a node with parameters that do not require a server to maintain any dynamic state for the node. The use of stateless DHCP is controlled by router advertisement (RA) messages multicasted by routers. The DHCPv6 client will invoke stateless DHCPv6 when it receives an appropriate RA. The DHCPv6 server will respond to a stateless DHCPv6 request with the appropriate configuration parameters, such as the DNS servers and domain search list options.

Client and Server Identification

Each DHCPv6 client and server is identified by a DHCP unique identifier (DUID). The DUID is carried in client identifier and server identifier options. The DUID is unique across all DHCP clients and servers, and it is stable for any specific client or server. DHCPv6 uses DUIDs based on link-layer addresses for both the client and server identifier. The device uses the MAC address from the lowest-numbered interface to form the DUID. The network interface is assumed to be permanently attached to the device.

When a DHCPv6 client requests two prefixes with the same DUID but with different IAIDs on two different interfaces, these prefixes are considered to be for two different clients, and the interface information is maintained for both.

Rapid Commit

The DHCPv6 client can obtain configuration parameters from a server either through a rapid two-message exchange (solicit, reply) or through a normal four-message exchange (solicit, advertise, request, reply). By default, the four-message exchange is used. When the rapid-commit option is enabled by both client and server, the two-message exchange is used.

DHCPv6 Client, Server, and Relay Functions

The DHCPv6 client, server, and relay functions are mutually exclusive on an interface. When one of these functions is enabled and a user tries to configure a different function on the same interface, one of the following messages is displayed: “Interface is in DHCP client mode,” “Interface is in DHCP server mode,” or “Interface is in DHCP relay mode.”

The following sections describe these functions:

Client Function

The DHCPv6 client function can be enabled on individual IPv6-enabled interfaces.

The DHCPv6 client can request and accept those configuration parameters that do not require a server to maintain any dynamic state for individual clients, such as DNS server addresses and domain search list options.

The DHCPv6 client can also request the delegation of prefixes. The prefixes acquired from a delegating router will be stored in a local IPv6 general prefix pool. The prefixes in the general prefix pool can then be referred to from other applications; for example, the general prefix pool can be used to number router downstream interfaces.

Server Selection

A DHCPv6 client builds a list of potential servers by sending a solicit message and by collecting advertise message replies from servers. These messages are ranked based on the preference value, and servers may add a preference option to their advertise messages explicitly stating their preference value. If the client needs to acquire prefixes from servers, only servers that have advertised prefixes are considered.

IAPD and IAID

An Identity Association for Prefix Delegation (IAPD) is a collection of prefixes assigned to a requesting router. A requesting router may have more than one IAPD; for example, one for each of its interfaces.

Each IAPD is identified by an IAID. The IAID is chosen by the requesting router and is unique among the IAPD IAIDs on the requesting router. IAIDs are made consistent across reboots by using information from the associated network interface, which is assumed to be permanently attached to the device.

Server Function

The DHCPv6 server function can be enabled on individual IPv6-enabled interfaces.

The DHCPv6 server can provide those configuration parameters that do not require the server to maintain any dynamic state for individual clients, such as DNS server addresses and domain search list options. The DHCPv6 server may be configured to perform prefix delegation.

All the configuration parameters for clients are independently configured into DHCPv6 configuration pools, which are stored in NVRAM. A configuration pool can be associated with a particular DHCPv6 server on an interface when it is started. Prefixes to be delegated to clients may be specified either as a list of preassigned prefixes for a particular client or as IPv6 local prefix pools that are also stored in NVRAM.

The list of manually configured prefixes or IPv6 local prefix pools can be referenced and used by DHCPv6 configuration pools.

The DHCPv6 server maintains an automatic binding table in memory to track the assignment of some configuration parameters, such as prefixes between the server and its clients. The automatic bindings can be stored permanently in the database agent, which can be, for example, a remote TFTP server or local NVRAM file system.

Configuration Information Pool

A DHCPv6 configuration information pool is a named entity that includes information about available configuration parameters and policies that control assignment of the parameters to clients from the pool. A pool is configured independently of the DHCPv6 service and is associated with the DHCPv6 service through the command-line interface (CLI).

Each configuration pool can contain the following configuration parameters and operational information:

- Prefix delegation information, which could include:
  - A prefix pool name and associated preferred and valid lifetimes
  - A list of available prefixes for a particular client and associated preferred and valid lifetimes
- A list of IPv6 addresses of DNS servers
- A domain search list, which is a string containing domain names for DNS resolution

DHCP for IPv6 Address Assignment

DHCPv6 enables DHCP servers to pass configuration parameters, such as IPv6 network addresses, to IPv6 clients. The DHCPv6 Individual Address Assignment feature manages nonduplicate address assignment in the correct prefix based on the network where the host is connected. Assigned addresses can be from one or multiple prefix pools. Additional options, such as the default domain and DNS name-server address, can be passed back to the client. Address pools can be assigned for use on a specific interface or on multiple interfaces, or the server can automatically find the appropriate pool.

Prefix Assignment

A prefix-delegating router (DHCPv6 server) selects prefixes to be assigned to a requesting router (DHCPv6 client) upon receiving a request from the client. The server can select prefixes for a requesting client using static assignment and dynamic assignment mechanisms. Administrators can manually configure a list of prefixes and associated preferred and valid lifetimes for an IAPD of a specific client that is identified by its DUID.

When the delegating router receives a request from a client, it checks if there is a static binding configured for the IAPD in the client's message. If a static binding is present, the prefixes in the binding are returned to the client. If no such a binding is found, the server attempts to assign prefixes for the client from other sources.

The Cisco IOS DHCPv6 server can assign prefixes dynamically from an IPv6 local prefix pool. When the server receives a prefix request from a client, it attempts to obtain unassigned prefixes from the pool. After the client releases the previously assigned prefixes, the server returns them to the pool for reassignment.

An IPv6 prefix delegating router can also select prefixes for a requesting router based on an external authority such as a RADIUS server using the Framed-IPv6-Prefix attribute. For more information on this feature, see the Implementing ADSL and Deploying Dial Access for IPv6 module.

Automatic Binding

Each DHCPv6 configuration pool has an associated binding table. The binding table contains the records about all the prefixes in the configuration pool that have been explicitly delegated to clients. Each entry in the binding table contains the following information:

- Client DUID
- Client IPv6 address
- A list of IAPDs associated with the client
- A list of prefixes delegated to each IAPD
- Preferred and valid lifetimes for each prefix
- The configuration pool to which this binding table belongs
- The network interface on which the server that is using the pool is running

A binding table entry is automatically created whenever a prefix is delegated to a client from the configuration pool, and it is updated when the client renews, rebinds, or confirms the prefix delegation. A binding table entry is deleted when the client releases all the prefixes in the binding voluntarily, all prefixes' valid lifetimes have expired, or administrators run the **clear ipv6 dhcp binding** command.

Binding Database

Each permanent storage to which the binding database is saved is called the database agent. A database agent can be a remote host such as an FTP server or a local file system such as NVRAM.

The automatic bindings are maintained in RAM and can be saved to some permanent storage so that the information about configuration such as prefixes assigned to clients is not lost after a system reload or power down. The bindings are stored as text records for easy maintenance. Each record contains the following information:

- DHCPv6 pool name from which the configuration was assigned to the client
- Interface identifier from which the client requests were received
- The client IPv6 address
- The client DUID
- IAID of the IAPD
- Prefix delegated to the client
- The prefix length
- The prefix preferred lifetime in seconds
- The prefix valid lifetime in seconds
- The prefix expiration time stamp
- Optional local prefix pool name from which the prefix was assigned

At the beginning of the file, before the text records, a time stamp records the time when the database is written and a version number, which helps differentiate between newer and older databases. At the end of the file, after the text records, the text string "*end*" is stored to detect file truncation.

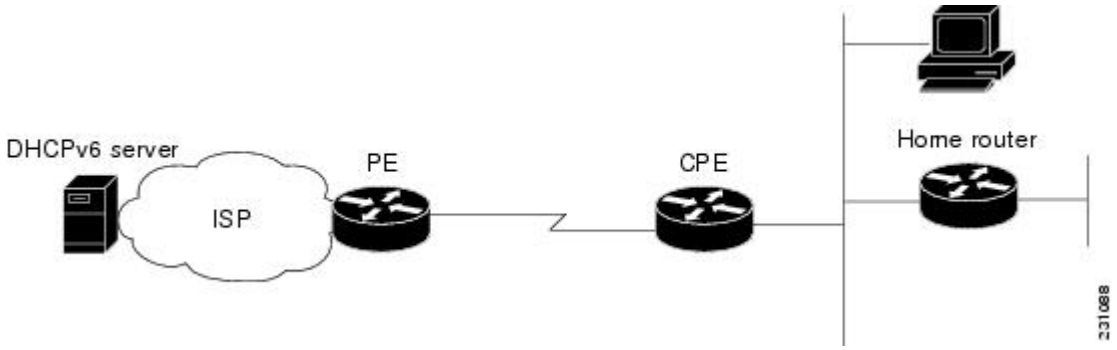
The permanent storage to which the binding database is saved is called the database agent. Database agents include FTP and TFTP servers, RCP, flash file system, and NVRAM.

DHCPv6 Server Stateless Autoconfiguration

Hierarchical DHCPv6 for stateless configuration parameters allows a stateless or stateful DHCPv6 client to export configuration parameters (DHCPv6 options) to a local DHCPv6 server pool. The local DHCPv6 server can then provide the imported configuration parameters to other DHCPv6 clients.

The figure below shows a typical broadband deployment.

Figure 20 Broadband Topology



The CPE interface toward the PE can be a stateless or stateful DHCPv6 client. In either case, the ISP-side DHCPv6 server may provide configuration parameters such as DNS server addresses, domain names, and Simple Network Time Protocol (SNTP) servers to the DHCP client on the CPE. These information can be specific to an ISP and may change.

In addition to being a DHCPv6 client (for example, toward the ISP), the CPE may act as a DHCPv6 server to the home network. For example, Neighbor Discovery followed by stateless or stateful DHCPv6 can occur on the link between CPE and the home devices (for example, the home router or PC). In some cases, the information to be provided to the home network is the same information obtained from the ISP-side DHCPv6 server. Because this information can be dynamically changed, it cannot be hard-configured in the CPE's configuration. Therefore, the DHCPv6 component on the CPE allows automatic importing of configuration parameters from the DHCPv6 client to the DHCPv6 server pool.

DHCPv6 provides support of the options for IPv6 on the server described in the following sections:

Information Refresh Server Option

The DHCPv6 information refresh option can specify an upper boundary for the length of time a client should wait before refreshing information retrieved from DHCPv6. This option is used with stateless DHCPv6, because there are no addresses or other entities with lifetimes that can tell the client when to contact the DHCPv6 server to refresh its configuration.

NIS- and NIS+-Related Server Options

Users can configure the network information service (NIS) or NIS plus (NIS+) address or domain name of a DHCPv6 server using NIS- and NIS+-related options, and then import that information to the DHCPv6 client.

SIP Server Options

Session initiation protocol (SIP) server options contain either a list of domain names or IPv6 addresses that can be mapped to one or more SIP outbound proxy servers. One option carries a list of domain names, and the other option carries a list of 128-bit IPv6 addresses.

SIP is an application-layer control protocol that can establish, modify and terminate multimedia sessions or calls. A SIP system has several logical components: user agents, proxy servers, redirect servers, and registrars. User agents may contain SIP clients; proxy servers always contain SIP clients.

SNTP Server Option

The SNTP server option provides a list of one or more IPv6 addresses of SNTP servers available to the client for synchronization. The clients use these SNTP servers to synchronize their system time to that of the standard time servers. The server may list the SNTP servers in decreasing order of preference, but clients must treat the list of SNTP servers as an ordered list.

DHCP Relay Agent

A DHCP relay agent, which may reside on the client's link, is used to relay messages between the client and server. DHCP relay agent operation is transparent to the client. A client locates a DHCP server using a reserved, link-scoped multicast address. Therefore, it is a requirement for direct communication between the client and the server that the client and the server be attached to the same link. However, in some situations in which ease of management, economy, or scalability is a concern, it is desirable to allow a DHCP client to send a message to a DHCP server that is not connected to the same link.

DHCPv6 Relay Agent Notification for Prefix Delegation

DHCPv6 relay agent notification for prefix delegation allows the router working as a DHCPv6 relay agent to find prefix delegation options by reviewing the contents of a DHCPv6 RELAY-REPLY packet that is being relayed by the relay agent to the client. When a prefix delegation option is found by the relay agent, the relay agent extracts the information about the prefix being delegated and inserts an IPv6 static route matching the prefix delegation information onto the relay agent. Future packets destined to that prefix via relay will be forwarded based on the information contained in the prefix delegation. The IPv6 static route is then left in the routing table until the prefix delegation lease time expires or the relay agent receives a release packet from the client releasing the prefix delegation.

No user configuration is required for this feature. Static route management is done automatically by the relay agent.

The IPv6 routes are added when the relay agent relays a RELAY-REPLY packet, and the IPv6 routes are deleted when the prefix delegation lease time expires or the relay agent receives a release message. An IPv6 static route in the routing table of the relay agent can be updated when the prefix delegation lease time is extended.

This feature leaves a static IPv6 route on the routing table of the relay agent. This registered IPv6 address allows unicast reverse packet forwarding (uRPF) to work by allowing the router doing the reverse lookup to confirm that the IPv6 address on the relay agent is not malformed or spoofed. The static route left in the routing table of the relay agent can be redistributed to other routing protocols to advertise the subnets to other nodes. The static routes will be removed when an DHCP_DECLINE message is sent by the client.

DHCPv6 Bulk-Lease Query

DHCPv6 supports bulk-lease query that allows a client to request information about DHCPv6 bindings. This functionality adds new query types and allows the bulk transfer of DHCPv6 binding data through TCP.

Bulk-lease query is enabled by default if the DHCPv6 relay agent is enabled. Bulk-lease query is triggered at the relay agent startup to retrieve binding information lost because of a reload. If a DHCPv6 relay destination is configured on an interface, bulk-lease query is performed by the IPv6 address of the interface on which DHCPv6 relay is enabled. Bulk-lease query is a separate process from the relay agent process.

DHCPv6 Relay Chaining

DHCPv6 messages can be relayed through multiple relay agents. This configuration is called *relay chaining*. Such a configuration can be supported only when each relay agent adds certain information to

DHCPv6 messages before relaying them. The additional information helps in relaying the DHCPv6 reply back to the DHCPv6 client through the same path.

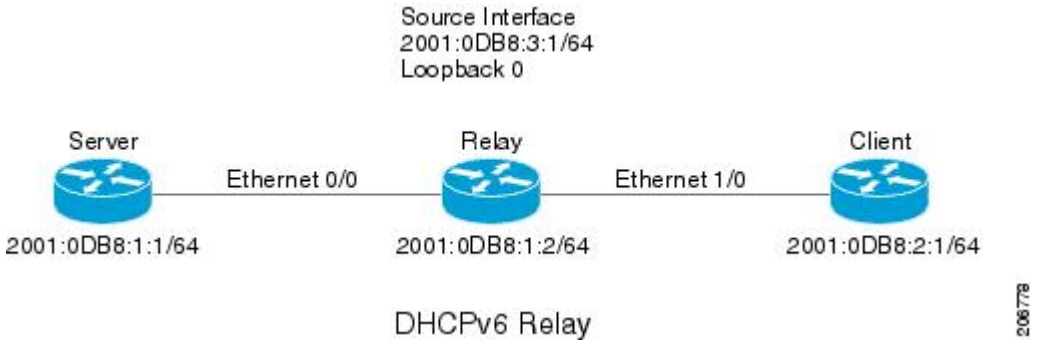
The delegated IPv6 prefix must be routable in order to be useful. The actual DHCPv6 Prefix Delegation (PD) client may not be permitted to inject routes into the delegating network. In service-provider (SP) networks, for example, an edge router typically acts as a DHCPv6 relay agent, and this edge router often has the responsibility to maintain routes within the SP network for clients' PD bindings. In the event that DHCPv6 requests and responses are relayed through a chain of DHCPv6 relays, there may be a need to introduce appropriate routes (particularly with DHCPv6 PD) in the Forwarding Information Base (FIB) so that routing is handled transparently.

DHCPv6 Relay Source Configuration

The DHCPv6 server sends its replies to the source address of relayed messages. Normally, a DHCPv6 relay uses the address of the server-facing interface used to send messages as the source. However, in some networks, it may be desirable to configure a more stable address (such as a loopback interface) and have the relay use that interface as the source address of relayed messages. The DHCPv6 Relay Source Configuration feature provides this capability.

The figure below shows a simple network with a single client, relay, and server. The relay and server communicate over 2001:DB8:1::/64, and the relay has a client-facing interface on 2001:DB8:2::/64. The relay also has a loopback interface configured with address 2001:DB8:3:1/64.

Figure 21 DHCPv6 Relay Source Configuration—Simple Network



When the relay receives a request from the client, the relay includes an address from the client-facing interface (Ethernet 1/0) in the link-address field of a relay-forward message. This address is used by the server to select an address pool. The relay then sends the relay-forward message toward the server. By default, the address of the server-facing (Ethernet 0/0) interface is used as the IPv6 source, and the server will send any reply to that address.

If the relay source interface is explicitly configured, the relay will use that interface's primary IPv6 address as the IPv6 source for messages it forwards. For example, configuring Loopback 0 as the source would cause the relay to use 2001:DB8:3:1/64 as the IPv6 source address for messages relayed toward the server.

DHCPv6 Relay SSO and ISSU

In specific Cisco networking devices that support dual route processors (RPs), stateful switchover (SSO) takes advantage of RP redundancy to increase network availability. The feature establishes one of the RPs as the active processor while the other RP is designated as the standby processor, and then synchronizing critical state information between them. Following an initial synchronization between the two processors, SSO dynamically maintains RP state information between them.

The Cisco IOS In Service Software Upgrade (ISSU) process allows Cisco IOS software to be updated or otherwise modified while packet forwarding continues. In most networks, planned software upgrades are a significant cause of downtime. ISSU allows the Cisco IOS software to be modified while packet forwarding continues. This increases network availability and reduces downtime caused by planned software upgrades.

The SSO and the ISSU use redundant hardware, with the active and standby RP each running an instance of the DHCPv6 relay agent. Both instances exchange run-time state data.

For further information about SSO and ISSU, see the "Stateful Switchover" and the "Cisco IOS In Service Software Upgrade Process" modules respectively, in the *Cisco IOS High Availability Configuration Guide*.

DHCPv6 Server and Relay—MPLS VPN Support

To facilitate managed central services in a Multiprotocol Label Switching (MPLS)-based network, DHCPv6 must be made MPLS-aware so a single resource can be used to serve multiple virtual private networks (VPNs) instead of dedicating a resource to a single VPN.

The DHCPv6 server implementation of MPLS VPN support allows a per-pool configuration so DHCPv6 pools can be associated with a VPN routing and forwarding (VRF) instance. The DHCPv6 server differentiates clients from various VRFs and assigns an IPv6 prefix accordingly from the respective VRF pools. Meanwhile, the DHCPv6 bindings store clients' VRF information.

The DHCPv6 relay implementation allows the configuration of the destination VRF instance to which the relay messages will be forwarded. The relay adds the client's VPN information while forwarding the client's DHCPv6 requests toward the server, and the relay then processes the client's VPN information in reply packets from server.

The relay adds IPv6 static routes for delegated prefixes in corresponding clients' VRF, and the relay's high availability (HA) functionality synchronizes the VRF information while synchronizing static routes created by the relay process.

The DHCPv6 relay and server VRF-aware features are disabled by default for backward compatibility.

How to Implement DHCP for IPv6

Configuring the DHCPv6 Server Function

The tasks in the following sections explain how to configure DHCPv6 server function:

Configuring the DHCPv6 Configuration Pool

Perform this task to create and configure the DHCPv6 configuration pool and associate the pool with a server on an interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **domain-name** *domain*
5. **dns-server** *ipv6-address*
6. **prefix-delegation** *ipv6-prefix / prefix-length client-duid* [**iaid** *iaid*] [*lifetime*]
7. **prefix-delegation pool** *poolname* [**lifetime** *valid-lifetime preferred-lifetime*]
8. **exit**
9. **interface** *type number*
10. **ipv6 dhcp server** *poolname* [**rapid-commit**] [**preference** *value*] [**allow-hint**]
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Router(config)# ipv6 dhcp pool pool1`	Configures a DHCPv6 configuration information pool and enters DHCPv6 pool configuration mode.

	Command or Action	Purpose
Step 4	**domain-name** *domain* **Example:** `Router(config-dhcp)# domain-name example.com`	Configures a domain name for a DHCPv6 client.
Step 5	**dns-server** *ipv6-address* **Example:** `Router(config-dhcp)# dns-server 2001:DB8:3000:3000::42`	Specifies the DNS IPv6 servers available to a DHCPv6 client.
Step 6	**prefix-delegation** *ipv6-prefix* / *prefix-length client-duid* [**iaid** *iaid*] [*lifetime*] **Example:** `Router(config-dhcp)# prefix-delegation 2001:DB8:1263::/48 0005000400F1A4D070D03`	Specifies a manually configured numeric prefix to be delegated to a specified client's IAPD.
Step 7	**prefix-delegation pool** *poolname* [**lifetime** *valid-lifetime preferred-lifetime*] **Example:** `Router(config-dhcp)# prefix-delegation pool pool1 lifetime 1800 60`	Specifies a named IPv6 local prefix pool from which prefixes are delegated to DHCPv6 clients.
Step 8	**exit** **Example:** `Router(config-dhcp)# exit`	Exits DHCPv6 pool configuration mode configuration mode, and returns the router to global configuration mode.
Step 9	**interface** *type number* **Example:** `Router(config)# interface serial 3`	Specifies an interface type and number, and enters interface configuration mode.
Step 10	**ipv6 dhcp server** *poolname* [**rapid-commit**] [**preference** *value*] [**allow-hint**] **Example:** `Router(config-if)# ipv6 dhcp server pool1`	Enables DHCPv6 on an interface.

	Command or Action	Purpose
Step 11	**end** **Example:** `Router(config-if)# end`	Returns to privileged EXEC mode.

Configuring a Binding Database Agent for the Server Function

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp database** *agent* [**write-delay** *seconds*] [**timeout** *seconds*]
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp database** *agent* [**write-delay** *seconds*] [**timeout** *seconds*] **Example:** `Router(config)# ipv6 dhcp database tftp://10.0.0.1/dhcp-binding`	Specifies DHCPv6 binding database agent parameters.
Step 4	**end** **Example:** `Router(config)# end`	Returns to privileged EXEC mode.

Configuring the DHCPv6 Client Function

General prefixes can be defined dynamically from a prefix received by a DHCPv6 prefix delegation client. The delegated prefix is stored in a general prefix.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 dhcp client pd** {*prefix-name* | **hint** *ipv6-prefix*} [**rapid-commit**]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface fastethernet 0/0/0`	Specifies an interface type and number, and enters interface configuration mode.
Step 4	**ipv6 dhcp client pd** {*prefix-name* | **hint** *ipv6-prefix*} [**rapid-commit**] **Example:** `Router(config-if)# ipv6 dhcp client pd dhcp-prefix`	Enables the DHCPv6 client process and enables a request for prefix delegation through a specified interface.
Step 5	**end** **Example:** `Router(config-if)# end`	Returns to privileged EXEC mode.

Configuring the DHCPv6 Relay Agent

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 dhcp relay destination** *ipv6-address* [*interface-type interface-number*]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface ethernet 4/2`	Specifies an interface type and number, and enters interface configuration mode.
Step 4	**ipv6 dhcp relay destination** *ipv6-address* [*interface-type interface-number*] **Example:** `Router(config-if) ipv6 dhcp relay destination FE80::250:A2FF:FEBF:A056 ethernet 4/3`	Specifies a destination address to which client packets are forwarded and enables DHCPv6 relay service on the interface.
Step 5	**end** **Example:** `Router(config-if)# end`	Returns to privileged EXEC mode.

Configuring Route Addition for Relay and Server

To enable route addition by DHCPv6 relay and server for the delegated prefix, use the **ipv6 dhcp iapd-route-add** command in global configuration mode.

To add routes for individually assigned IPv6 addresses on the relay or server, use the **ipv6 dhcp iana-route-add** command in global configuration mode

Configuring a DHCPv6 Relay Source

Perform the following tasks to configure a DHCPv6 relay source:

Restrictions for Configuring a DHCPv6 Relay Source

- If the configured interface is shut down, or if all of its IPv6 addresses are removed, the relay will revert to its standard behavior.
- The command line interface (CLI) will report an error if the user attempts to specify an interface that has no IPv6 addresses configured.
- The interface configuration takes precedence over the global configuration if both have been configured.

Configuring a DHCPv6 Relay Source on an Interface

Perform this task to configure an interface to use as the source when relaying messages.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 dhcp relay source-interface** *interface-type interface-number*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface loopback 0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**ipv6 dhcp relay source-interface** *interface-type interface-number* **Example:** `Router(config-if)# ipv6 dhcp relay source-interface loopback 0`	Configures an interface to use as the source when relaying messages received on this interface.
Step 5	**end** **Example:** `Router(config-if)# end`	Returns to privileged EXEC mode.

Configuring a DHCPv6 Relay Source Globally

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp-relay source-interface** *interface-type interface-number*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp-relay source-interface** *interface-type interface-number* **Example:** `Router(config)# ipv6 dhcp-relay source-interface loopback 0`	Configures an interface to use as the source when relaying messages.
Step 4	**end** **Example:** `Router(config)# end`	Returns to privileged EXEC mode.

Configuring DHCPv6 Bulk-Lease Query Parameters

The DHCPv6 Bulk-Lease Query feature is enabled automatically when the DHCPv6 relay agent is enabled.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp-relay bulk-lease** {**data-timeout** *seconds* | **retry** *number*} [**disable**]
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ipv6 dhcp-relay bulk-lease** {**data-timeout** *seconds* | **retry** *number*} [**disable**] **Example:** `Router(config)# ipv6 dhcp-relay bulk-lease data-timeout 60`	Configures bulk-lease query parameters.
Step 4	**end** **Example:** `Router(config)# end`	Returns to privileged EXEC mode.

Configuring DHCP for IPv6 Address Assignment

Perform the following tasks to configure DHCPv6 address assignment:

Prerequisites for Configuring DHCPv6 Address Assignment

By default, no DHCPv6 features are configured on the router.

When configuring DHCPv6 address assignment, remember that the specified interface must be one of these Layer 3 interfaces:

- Switch Virtual Interface (SVI): a VLAN interface created by using the **interface vlan** *vlan-id* command.
- EtherChannel port channel in Layer 3 mode: a port-channel logical interface created by using the **interface port-channel port-channel-number** command.

Enabling the DHCPv6 Server Function on an Interface

Perform this task to enable the DHCPv6 server function on an interface. Note that to delete a DHCPv6 pool, you must use the **no ipv6 dhcp pool** *poolname* global configuration command. Use the **no** form of the DHCP pool configuration mode commands to change the DHCPv6 pool characteristics. To disable the DHCPv6 server function on an interface, use the **no ipv6 dhcp server** interface configuration command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **address prefix** *ipv6-prefix* [**lifetime** {*valid-lifetime preferred-lifetime* | **infinite**}]
5. **link-address** *ipv6-prefix*
6. **vendor-specific** *vendor-id*
7. **suboption** *number* {**address** *ipv6-address* | **ascii** ascii-string | **hex** *hex-string*}
8. **exit**
9. **exit**
10. **interface** *type number*
11. **ipv6 dhcp server** [*poolname* | **automatic**] [**rapid-commit**] [**preference** *value*] [**allow-hint**]
12. **end**
13. Do one of the following:
    - **show ipv6 dhcp pool**
    - **show ipv6 dhcp interface**
14. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router(config)# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Router(config)# ipv6 dhcp pool engineering`	Enters DHCP pool configuration mode, and defines the name of the IPv6 DHCP pool.

	Command or Action	Purpose
Step 4	**address prefix** *ipv6-prefix* [**lifetime** {*valid-lifetime preferred-lifetime* | **infinite**}] **Example:** `Router(config-dhcpv6)# address prefix 2001:1000::0/64 lifetime infinite`	(Optional) Specifies an address prefix for address assignment. • This address must be in hexadecimal, using 16-bit values between colons. • **lifetime** *valid-lifetime preferred-lifetime*—Specifies a time interval (in seconds) that an IPv6 address prefix remains in the valid state.
Step 5	**link-address** *ipv6-prefix* **Example:** `Router(config-dhcpv6)# link-address 2001:1001::0/64`	(Optional) Specifies a link-address IPv6 prefix. • When an address on the incoming interface or a link address in the packet matches the specified IPv6 prefix, the server uses the configuration information pool.
Step 6	**vendor-specific** *vendor-id* **Example:** `Router(config-dhcpv6)# vendor-specific 9`	(Optional) Enters vendor-specific configuration mode with the vendor-specific identification number.
Step 7	**suboption** *number* {**address** *ipv6-address* | **ascii** ascii-string | **hex** *hex-string*} **Example:** `Router(config-dhcpv6-vs)# suboption 1 address 1000:235D::1`	(Optional) Enters a vendor-specific suboption number.
Step 8	**exit** **Example:** `Router(config-dhcpv6-vs)# exit`	Returns to DHCP pool configuration mode.
Step 9	**exit** **Example:** `Router(config-dhcpv6)# exit`	Returns to global configuration mode.
Step 10	**interface** *type number* **Example:** `Router(config)# interface fastethernet 0/0`	Enters interface configuration mode, and specifies the interface to configure.

	Command or Action	Purpose
Step 11	**ipv6 dhcp server** [*poolname* | **automatic**] [**rapid-commit**] [**preference** *value*] [**allow-hint**] **Example:** `Router(config-if)# ipv6 address dhcp server rapid-commit`	Enables DHCPv6 server function on an interface.
Step 12	**end** **Example:** `Router(config-if)# end`	Returns to privileged EXEC mode.

Step 13	Do one of the following:
	• **show ipv6 dhcp pool**
	• **show ipv6 dhcp interface**
	
	Example:
	`Router# show ipv6 dhcp pool`
	
	Example:
	
	Example:
	`Router# show ipv6 dhcp interface`
		Verifies DHCPv6 pool configuration or verifies that the DHCPv6 server function is enabled on an interface.

Step 14	**copy running-config startup-config** **Example:** `Router# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Enabling the DHCPv6 Client Function on an Interface

Perform this task to enable the DHCPv6 client function on an interface. To disable the DHCPv6 client function, use the **no ipv6 address dhcp** interface configuration command. To remove the DHCPv6 client request, use the **no ipv6 address dhcp client request vendor** interface configuration command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 address dhcp** [**rapid-commit**]
5. **ipv6 address dhcp client request vendor**
6. **end**
7. **show ipv6 dhcp interface**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface fastethernet 0/0`	Enters interface configuration mode, and specifies the interface to configure.
Step 4	**ipv6 address dhcp** [**rapid-commit**] **Example:** `Router(config-if)# ipv6 address dhcp rapid-commit`	Enables the interface to acquire an IPv6 address from the DHCPv6 server.
Step 5	**ipv6 address dhcp client request vendor** **Example:** `Router(config-if)# ipv6 dhcp client request vendor-specific`	(Optional) Enables the interface to request the vendor-specific option.

	Command or Action	Purpose
Step 6	**end** **Example:** `Router(config-if)# end`	Returns to privileged EXEC mode.
Step 7	**show ipv6 dhcp interface** **Example:** `Router# show ipv6 dhcp interface`	Verifies that the DHCPv6 client is enabled on an interface.

Configuring the Stateless DHCPv6 Function

The server maintains no state related to clients; for example, no prefix pools and records of allocation are maintained. Therefore, this function is “stateless” DHCPv6.

Configuring the Stateless DHCPv6 Server

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **dns-server** *ipv6-address*
5. **domain-name** *domain*
6. **exit**
7. **interface** *type number*
8. **ipv6 dhcp server** *poolname* [**rapid-commit**] [**preference** *value*] [**allow-hint**]
9. **ipv6 nd other-config-flag**
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Router(config)# ipv6 dhcp pool dhcp-pool`	Configures a DHCPv6 configuration information pool and enters DHCPv6 pool configuration mode.
Step 4	**dns-server** *ipv6-address* **Example:** `Router(config-dhcp) dns-server 2001:DB8:3000:3000::42`	Specifies the DNS IPv6 servers available to a DHCPv6 client.
Step 5	**domain-name** *domain* **Example:** `Router(config-dhcp)# domain-name domain1.com`	Configures a domain name for a DHCPv6 client.
Step 6	**exit** **Example:** `Router(config-dhcp)# exit`	Exits DHCPv6 pool configuration mode, and returns the router to global configuration mode.
Step 7	**interface** *type number* **Example:** `Router(config)# interface serial 3`	Specifies an interface type and number, and places the router in interface configuration mode.

	Command or Action	Purpose
Step 8	**ipv6 dhcp server** *poolname* [**rapid-commit**] [**preference** *value*] [**allow-hint**] **Example:** `Router(config-if)# ipv6 dhcp server dhcp-pool`	Enables DHCPv6 on an interface.
Step 9	**ipv6 nd other-config-flag** **Example:** `Router(config-if)# ipv6 nd other-config-flag`	Sets the “other stateful configuration” flag in IPv6 RAs.
Step 10	**end** **Example:** `Router(config-if)# end`	Returns to privileged EXEC mode.

Configuring the Stateless DHCPv6 Client

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 address autoconfig** [**default**]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Router(config)# interface serial 3`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**ipv6 address autoconfig** [**default**] **Example:** `Router(config-if)# ipv6 address autoconfig`	Enables automatic configuration of IPv6 addresses using stateless autoconfiguration on an interface and enables IPv6 processing on the interface.
Step 5	**end** **Example:** `Router(config-if)# end`	Returns to privileged EXEC mode.

Enabling Processing of Packets with Source Routing Header Options

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 source-route**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ipv6 source-route** **Example:** `Router(config)# ipv6 source-route`	Enables the processing of the IPv6 type 0 routing header.
Step 4	**end** **Example:** `Router(config)# end`	Returns to privileged EXEC mode.

Configuring the DHCPv6 Server Options

Configuring the Information Refresh Server Option

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **information refresh** {*days* [*hours minutes*] | **infinity**}
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Router(config)# ipv6 dhcp pool pool1`	Configures a DHCPv6 configuration information pool and enters DHCPv6 pool configuration mode.
Step 4	**information refresh** {*days* [*hours minutes*] | **infinity**} **Example:** `Router(config-dhcp)# information refresh 1 1 1`	Specifies the information refresh time to be sent to the client.
Step 5	**end** **Example:** `Router(config-dhcp)# end`	Returns to privileged EXEC mode.

Importing the Information Refresh Server Option

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **import information refresh**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Router(config)# ipv6 dhcp pool pool1`	Configures a DHCPv6 configuration information pool and enters DHCPv6 pool configuration mode.
Step 4	**import information refresh** **Example:** `Router(config-dhcp)# import information refresh`	Imports the information refresh time option to a DHCPv6 client.
Step 5	**end** **Example:** `Router(config-dhcp)# end`	Returns to privileged EXEC mode.

Configuring NIS- and NISP-Related Server Options

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **nis address** *ipv6-address*
5. **nis domain-name** *domain-name*
6. **nisp address** *ipv6-address*
7. **nisp domain-name** *domain-name*
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Router(config)# ipv6 dhcp pool pool1`	Configures a DHCPv6 configuration information pool and enters DHCPv6 pool configuration mode.
Step 4	**nis address** *ipv6-address* **Example:** `Router(config-dhcp)# nis address 2001:DB8:1000:1000::30`	Specifies the NIS address of an IPv6 server to be sent to the client.
Step 5	**nis domain-name** *domain-name* **Example:** `Router(config-dhcp)# nis domain-name domain1`	Enables a server to convey a client's NIS domain name information to the client.
Step 6	**nisp address** *ipv6-address* **Example:** `Router(config-dhcp)# nisp address 2001:DB8:3000:3000::42`	Specifies the NIS+ address of an IPv6 server to be sent to the DHCPv6 client.
Step 7	**nisp domain-name** *domain-name* **Example:** `Router(config-dhcp)# nisp domain-name domain2`	Enables a server to convey a client's NIS+ domain name information to the DHCPv6 client.

	Command or Action	Purpose
Step 8	**end** **Example:** `Router(config-dhcp)# end`	Returns to privileged EXEC mode.

Importing NIS- and NIS+-Related Server Options

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **import nis address**
5. **import nis domain-name**
6. **import nisp address**
7. **import nisp domain-name**
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Router(config)# ipv6 dhcp pool pool1`	Configures a DHCPv6 configuration information pool and enters DHCPv6 pool configuration mode.

	Command or Action	Purpose
Step 4	**import nis address** **Example:** `Router(config-dhcp)# import nis address`	Imports the NIS servers option to a DHCPv6 client.
Step 5	**import nis domain-name** **Example:** `Router(config-dhcp)# import nis domain-name`	Imports the NIS domain name option to a DHCPv6 client.
Step 6	**import nisp address** **Example:** `Router(config-dhcp)# import nisp address`	Imports the NISP address option to a DHCPv6 client.
Step 7	**import nisp domain-name** **Example:** `Router(config-dhcp)# import nisp domain-name`	Imports the NISP domain name option to a DHCPv6 client.
Step 8	**end** **Example:** `Router(config-dhcp)# end`	Returns to privileged EXEC mode.

Importing SIP Server Options

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **import sip address**
5. **import sip domain-name**
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Router(config)# ipv6 dhcp pool pool1`	Configures a DHCPv6 configuration information pool and enters DHCPv6 pool configuration mode.
Step 4	**import sip address** **Example:** `Router(config-dhcp)# import sip address`	Imports the SIP server IPv6 address list option to the outbound SIP proxy server.
Step 5	**import sip domain-name** **Example:** `Router(config-dhcp)# import sip domain-name`	Imports a SIP server domain-name list option to the outbound SIP proxy server.
Step 6	**end** **Example:** `Router(config-dhcp)# end`	Returns to privileged EXEC mode.

Configuring the SNTP Server

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **sntp address** *ipv6-address*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Router(config)# ipv6 dhcp pool pool1`	Configures a DHCPv6 configuration information pool and enters DHCPv6 pool configuration mode.
Step 4	**sntp address** *ipv6-address* **Example:** `Router(config-dhcp)# sntp address 2001:DB8:2000:2000::33`	Specifies the SNTP server list to be sent to the client.
Step 5	**end** **Example:** `Router(config-dhcp)# end`	Returns to privileged EXEC mode.

Importing the SNTP Server Option

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **import sntp address** *ipv6-address*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Router(config)# ipv6 dhcp pool pool1`	Configures a DHCPv6 configuration information pool and enters DHCPv6 pool configuration mode.
Step 4	**import sntp address** *ipv6-address* **Example:** `Router(config-dhcp)# import sntp address 2001:DB8:2000:2000::33`	Imports the SNTP server option to a DHCPv6 client.
Step 5	**end** **Example:** `Router(config-dhcp)# end`	Returns to privileged EXEC mode.

Importing Stateless DHCPv6 Server Options

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **import dns-server**
5. **import domain-name**
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Router(config)# ipv6 dhcp pool pool1`	Configures a DHCPv6 configuration information pool and enters DHCPv6 pool configuration mode.
Step 4	**import dns-server** **Example:** `Router(config-dhcp)# import dns-server`	Imports the DNS recursive name server option to a DHCPv6 client.
Step 5	**import domain-name** **Example:** `Router(config-dhcp)# import domain-name`	Imports the domain search list option to a DHCPv6 client.

	Command or Action	Purpose
Step 6	**end** **Example:** `Router(config-dhcp)# end`	Returns to privileged EXEC mode.

Defining a General Prefix with the DHCPv6 Prefix Delegation Client Function

Perform this task to configure the DHCPv6 client function on an interface and enable prefix delegation on an interface. The delegated prefix is stored in a general prefix.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 dhcp client pd** {*prefix-name* | **hint** *ipv6-prefix*} [**rapid-commit**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface ethernet 0/0`	Specifies an interface type and number, and enters interface configuration mode.

	Command or Action	Purpose
Step 4	**ipv6 dhcp client pd** {*prefix-name* | **hint** *ipv6-prefix*} [**rapid-commit**] **Example:** `Router(config-if)# ipv6 dhcp client pd dhcp-prefix`	Enables the DHCPv6 client process and enables a request for prefix delegation through a specified interface. • The delegated prefix is stored in the general prefix *prefix-name* argument.

Configuring a VRF-Aware Relay and Server for MPLS VPN Support

Configuring a VRF-Aware Relay

Note that you do not have to configure this feature on specified interfaces; if you want the feature to be enabled globally on the router only, perform steps 1, 2, and 3.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp-relay option vpn**
4. **interface** *type number*
5. **ipv6 dhcp relay option vpn**
6. **ipv6 dhcp relay destination** *ipv6-address* [*interface-type interface-number* | **vrf** *vrf-name* | **global**]
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ipv6 dhcp-relay option vpn** **Example:** `Router(config)# ipv6 dhcp-relay option vpn`	Enables the DHCP for IPv6 relay VRF-aware feature globally.
Step 4	**interface** *type number* **Example:** `Router(config)# interface ethernet 0/0`	Specifies an interface type and number, and enters interface configuration mode.
Step 5	**ipv6 dhcp relay option vpn** **Example:** `Router(config-if)# ipv6 dhcp relay option vpn`	Enables the DHCP for IPv6 relay VRF-aware feature on the specified interface. Enabling this command supersedes configuration using the **ipv6 dhcp-relay option vpn** command.
Step 6	**ipv6 dhcp relay destination** *ipv6-address* [*interface-type interface-number* | **vrf** *vrf-name* | **global**] **Example:** `Router(config-if)# ipv6 dhcp relay destination FE80::250:A2FF:FEBF:A056 ethernet 0/0`	Specifies a destination address to which client messages are forwarded.
Step 7	**end** **Example:** `Router(config-if)# end`	Returns to privileged EXEC mode.

Configuring a VRF-Aware Server

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 dhcp server vrf enable**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface ethernet 0/0`	Specifies an interface type and number, and enters interface configuration mode.
Step 4	**ipv6 dhcp server vrf enable** **Example:** `Router(config-if)# ipv6 dhcp server vrf enable`	Enables the DHCPv6 server VRF-aware feature on an interface.
Step 5	**end** **Example:** `Router(config-if)# end`	Returns to privileged EXEC mode.

Deleting Automatic Client Bindings from the DHCPv6 Binding Table

SUMMARY STEPS

1. **enable**
2. **clear ipv6 dhcp binding** [*ipv6-address*] [**vrf** *vrf-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ipv6 dhcp binding** [*ipv6-address*] [**vrf** *vrf-name*] **Example:** `Router# clear ipv6 dhcp binding`	Deletes automatic client bindings from the DHCPv6 binding table.

Troubleshooting DHCPv6

SUMMARY STEPS

1. **enable**
2. **debug ipv6 dhcp** [**detail**]
3. **debug ipv6 dhcp database**
4. **debug ipv6 dhcp relay**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**debug ipv6 dhcp** [**detail**] **Example:** `Router# debug ipv6 dhcp`	Enables debugging for DHCPv6.
Step 3	**debug ipv6 dhcp database** **Example:** `Router# debug ipv6 dhcp database`	Enables debugging for the DHCPv6 binding database.

	Command or Action	Purpose
Step 4	**debug ipv6 dhcp relay** **Example:** `Router# debug ipv6 dhcp relay`	Enables DHCPv6 relay agent debugging.

Verifying DHCPv6 Configuration and Operation

SUMMARY STEPS

1. **enable**
2. **show ipv6 dhcp**
3. **show ipv6 dhcp binding** [*ipv6-address*]
4. **show ipv6 dhcp database** [*agent-URL*]
5. **show ipv6 dhcp interface** [*type number*]
6. **show ipv6 dhcp pool** [*poolname*]
7. **show running-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ipv6 dhcp** **Example:** `Router# show ipv6 dhcp`	Displays the DUID on a specified device.
Step 3	**show ipv6 dhcp binding** [*ipv6-address*] **Example:** `Router# show ipv6 dhcp binding`	Displays automatic client bindings from the DHCPv6 database.

	Command or Action	Purpose
Step 4	**show ipv6 dhcp database** [*agent-URL*] **Example:** `Router# show ipv6 dhcp database`	Displays the DHCPv6 binding database agent information.
Step 5	**show ipv6 dhcp interface** [*type number*] **Example:** `Router# show ipv6 dhcp interface`	Displays DHCPv6 interface information.
Step 6	**show ipv6 dhcp pool** [*poolname*] **Example:** `Router# show ipv6 dhcp pool`	Displays DHCPv6 configuration pool information.
Step 7	**show running-config** **Example:** `Router# show running-config`	Displays the current configuration running on the router.

- Examples, page 223

Examples

Sample Output from the show ipv6 dhcp Command

The following example from the **show ipv6 dhcp** command shows the DUID of the device:

```
Router# show ipv6 dhcp
This device's DHCPv6 unique identifier(DUID): 000300010002FCA5DC1C
```

Sample Output from the show ipv6 dhcp binding Command

In the following example, the **show ipv6 dhcp binding** command shows information about two clients, including their DUIDs, IAPDs, prefixes, and preferred and valid lifetimes:

```
Router# show ipv6 dhcp binding
Client: FE80::202:FCFF:FEA5:DC39 (Ethernet2/1)
  DUID: 000300010002FCA5DC1C
  IA PD: IA ID 0x00040001, T1 0, T2 0
    Prefix: 3FFE:C00:C18:11::/68
            preferred lifetime 180, valid lifetime 12345
            expires at Nov 08 2002 02:24 PM (12320 seconds)
Client: FE80::202:FCFF:FEA5:C039 (Ethernet2/1)
  DUID: 000300010002FCA5C01C
  IA PD: IA ID 0x00040001, T1 0, T2 0
```

```
    Prefix: 3FFE:C00:C18:1::/72
            preferred lifetime 240, valid lifetime 54321
            expires at Nov 09 2002 02:02 AM (54246 seconds)
    Prefix: 3FFE:C00:C18:2::/72
            preferred lifetime 300, valid lifetime 54333
            expires at Nov 09 2002 02:03 AM (54258 seconds)
    Prefix: 3FFE:C00:C18:3::/72
            preferred lifetime 280, valid lifetime 51111
```

Sample Output from the show ipv6 dhcp database Command

In the following example, the **show ipv6 dhcp database** command provides information on the binding database agents TFTP, NVRAM, and flash:

```
Router# show ipv6 dhcp database
Database agent tftp://172.19.216.133/db.tftp:
  write delay: 69 seconds, transfer timeout: 300 seconds
  last written at Jan 09 2003 01:54 PM,
     write timer expires in 56 seconds
  last read at Jan 06 2003 05:41 PM
  successful read times 1
  failed read times 0
  successful write times 3172
  failed write times 2
Database agent nvram:/dhcpv6-binding:
  write delay: 60 seconds, transfer timeout: 300 seconds
  last written at Jan 09 2003 01:54 PM,
     write timer expires in 37 seconds
  last read at never
  successful read times 0
  failed read times 0
  successful write times 3325
  failed write times 0
Database agent flash:/dhcpv6-db:
  write delay: 82 seconds, transfer timeout: 3 seconds
  last written at Jan 09 2003 01:54 PM,
    write timer expires in 50 seconds
  last read at never
  successful read times 0
  failed read times 0
  successful write times 2220
  failed write times 614
```

Sample Output from the show ipv6 dhcp interface Command

The following is sample output from the **show ipv6 dhcp interface** command. In the first example, the command is used on a router that has an interface acting as a DHCPv6 server. In the second example, the command is used on a router that has an interface acting as a DHCPv6 client:

```
Router1# show ipv6 dhcp interface
Ethernet2/1 is in server mode
  Using pool: svr-p1
  Preference value: 20
  Rapid-Commit is disabled
Router2# show ipv6 dhcp interface
Ethernet2/1 is in client mode
  State is OPEN (1)
  List of known servers:
    Address: FE80::202:FCFF:FEA1:7439, DUID 000300010002FCA17400
    Preference: 20
      IA PD: IA ID 0x00040001, T1 120, T2 192
        Prefix: 3FFE:C00:C18:1::/72
                preferred lifetime 240, valid lifetime 54321
                expires at Nov 08 2002 09:10 AM (54319 seconds)
        Prefix: 3FFE:C00:C18:2::/72
                preferred lifetime 300, valid lifetime 54333
                expires at Nov 08 2002 09:11 AM (54331 seconds)
        Prefix: 3FFE:C00:C18:3::/72
                preferred lifetime 280, valid lifetime 51111
```

```
            expires at Nov 08 2002 08:17 AM (51109 seconds)
      DNS server: 2001:DB8:1001::1
      DNS server: 2001:DB8:1001::2
      Domain name: example1.net
      Domain name: example2.net
      Domain name: example3.net
    Prefix name is cli-p1
    Rapid-Commit is enabled
```

Sample Output from the show ipv6 dhcp pool Command

In the following example, the **show ipv6 dhcp pool** command provides information on the configuration pool named svr-p1, including the static bindings, prefix information, the DNS server, and the domain names found in the svr-p1 pool:

```
Router# show ipv6 dhcp pool

DHCPv6 pool: svr-p1
  Static bindings:
    Binding for client 000300010002FCA5C01C
      IA PD: IA ID 00040002,
        Prefix: 3FFE:C00:C18:3::/72
                preferred lifetime 604800, valid lifetime 2592000
      IA PD: IA ID not specified; being used by 00040001
        Prefix: 3FFE:C00:C18:1::/72
                preferred lifetime 240, valid lifetime 54321
        Prefix: 3FFE:C00:C18:2::/72
                preferred lifetime 300, valid lifetime 54333
        Prefix: 3FFE:C00:C18:3::/72
                preferred lifetime 280, valid lifetime 51111
  Prefix from pool: local-p1, Valid lifetime 12345, Preferred lifetime 180
  DNS server: 2001:DB8:1001::1
  DNS server: 2001:DB8:1001::2
  Domain name: example1.net
  Domain name: example2.net
  Domain name: example3.net
Active clients: 2
Current configuration : 22324 bytes
!
! Last configuration change at 14:59:38 PST Tue Jan 16 2001
! NVRAM config last updated at 04:25:39 PST Tue Jan 16 2001 by bird
!
hostname Router
!
ip cef
ipv6 unicast-routing
ipv6 cef
ipv6 cef accounting prefix-length
!
!
interface Ethernet0
 ip address 10.4.9.11 255.0.0.0
 media-type 10BaseT
 ipv6 address 2001:DB8:C18:1::/64 eui-64
```

Configuration Examples for Implementing DHCPv6

Example: Configuring the DHCPv6 Server Function

DHCPv6 clients are connected to the DHCPv6 server on Ethernet interface 0/0. The server is configured to use parameters from the DHCP pool called dhcp-pool. This pool provides clients with the IPv6 address of a DNS server and the domain name to be used. It also specifies that prefixes can be delegated from the prefix pool called client-prefix-pool1. The prefixes delegated will have valid and preferred lifetimes of 1800 and 600 seconds respectively. The prefix pool named client-prefix-pool1 has a prefix of length /40 from which it will delegate (sub) prefixes of length /48.

```
ipv6 dhcp pool dhcp-pool
 prefix-delegation pool client-prefix-pool1 lifetime 1800 600
 dns-server 2001:DB8:3000:3000::42
 domain-name example.com
!
interface Ethernet0/0
 description downlink to clients
 ipv6 address FEC0:240:104:2001::139/64
 ipv6 dhcp server dhcp-pool
!
ipv6 local pool client-prefix-pool1 2001:DB8:1200::/40 48
```

The following example from the show ipv6 dhcp command shows the DUID of the device:

```
Router# show ipv6 dhcp

This device's DHCPv6 unique identifier(DUID): 000300010002FCA5DC1C
```

In the following example, the show ipv6 dhcp binding command shows information about two clients, including their DUIDs, IAPDs, prefixes, and preferred and valid lifetimes:

```
Router# show ipv6 dhcp binding

Client: FE80::202:FCFF:FEA5:DC39 (Ethernet2/1)
  DUID: 000300010002FCA5DC1C
  IA PD: IA ID 0x00040001, T1 0, T2 0
    Prefix: 3FFE:C00:C18:11::/68
            preferred lifetime 180, valid lifetime 12345
            expires at Nov 08 2002 02:24 PM (12320 seconds)
Client: FE80::202:FCFF:FEA5:C039 (Ethernet2/1)
  DUID: 000300010002FCA5C01C
  IA PD: IA ID 0x00040001, T1 0, T2 0
    Prefix: 3FFE:C00:C18:1::/72
            preferred lifetime 240, valid lifetime 54321
            expires at Nov 09 2002 02:02 AM (54246 seconds)
    Prefix: 3FFE:C00:C18:2::/72
            preferred lifetime 300, valid lifetime 54333
            expires at Nov 09 2002 02:03 AM (54258 seconds)
    Prefix: 3FFE:C00:C18:3::/72
            preferred lifetime 280, valid lifetime 51111
```

In the following example, the show ipv6 dhcp database command provides information on the binding database agents TFTP, NVRAM, and flash:

```
Router# show ipv6 dhcp database

Database agent tftp://172.19.216.133/db.tftp:
  write delay: 69 seconds, transfer timeout: 300 seconds
  last written at Jan 09 2003 01:54 PM,
     write timer expires in 56 seconds
  last read at Jan 06 2003 05:41 PM
  successful read times 1
  failed read times 0
  successful write times 3172
  failed write times 2
Database agent nvram:/dhcpv6-binding:
  write delay: 60 seconds, transfer timeout: 300 seconds
  last written at Jan 09 2003 01:54 PM,
     write timer expires in 37 seconds
  last read at never
```

```
  successful read times 0
  failed read times 0
  successful write times 3325
  failed write times 0
Database agent flash:/dhcpv6-db:
  write delay: 82 seconds, transfer timeout: 3 seconds
  last written at Jan 09 2003 01:54 PM,
    write timer expires in 50 seconds
  last read at never
  successful read times 0
  failed read times 0
  successful write times 2220
  failed write times 614
```

Example: Configuring the DHCPv6 Client Function

This DHCPv6 client has three interfaces: Ethernet interface 0/0 is the upstream link to a service provider, which has a DHCPv6 server function enabled. The Fast Ethernet interfaces 0/0 and 0/1 are links to local networks.

The upstream interface, Ethernet interface 0/0, has the DHCPv6 client function enabled. Prefixes delegated by the provider are stored in the general prefix called prefix-from-provider.

The local networks, Fast Ethernet interfaces 0/0 and 0/1, both assign interface addresses based on the general prefix called prefix-from-provider. The bits on the left of the addresses come from the general prefix, and the bits on the right of the addresses are specified statically.

```
interface Ethernet 0/0
 description uplink to provider DHCP IPv6 server
 ipv6 dhcp client pd prefix-from-provider
!
interface FastEthernet 0/0
 description local network 0
 ipv6 address prefix-from-provider ::5:0:0:0:100/64
!
interface FastEthernet 0/1
 description local network 1
 ipv6 address prefix-from-provider ::6:0:0:0:100/64
```

Example: Configuring a Database Agent for the Server Function

The DHCPv6 server is configured to store table bindings to the file named dhcp-binding on the server at address 10.0.0.1 using the TFTP protocol. The bindings are saved every 120 seconds.

```
ipv6 dhcp database tftp://10.0.0.1/dhcp-binding write-delay 120
```

The following example shows how to specify DHCP for IPv6 binding database agent parameters and store binding entries in bootflash:

```
ipv6 dhcp database bootflash
```

Example: Configuring DHCP for IPv6 Address Assignment

The following example shows how to configure a pool called engineering with an IPv6 address prefix:

```
ipv6 dhcp pool engineering
address prefix 2001:1000::0/64 lifetime infinite
```

The following example shows how to configure A pool called testgroup with three link addresses and an IPv6 address prefix:

```
ipv6 dhcp pool testgroup
```

```
link-address 2001:1001::0/64
link-address 2001:1002::0/64
link-address 2001:2000::0/48
address prefix 2001:1000::0/64 lifetime infinite
end
```

The following example shows how to configure a pool called 350 with vendor-specific options:

```
ipv6 dhcp pool 350
address prefix 2001:1000::0/64 lifetime infinite
vendor-specific 9
suboption 1 address 1000:235D::1
suboption 2 ascii "IP-Phone"
end
```

Example: Configuring the Stateless DHCPv6 Function

The following example shows how to use the DHCPv6 function to configure clients with information about the name lookup system. The server is configured with a DHCP pool, which contains the name lookup information that is to be passed to clients. It does not need to contain a prefix pool. This DHCP pool is attached to the access link to customers (Ethernet0/0) using the **ipv6 dhcp server** command. The access link also has the **ipv6 nd other-config-flag** command enabled. RA messages sent from this interface will inform clients that they should use DHCPv6 for "other" (for example, nonaddress) configuration information.

```
ipv6 dhcp pool dhcp-pool
 dns-server 2001:DB8:A:B::1
 dns-server 2001:DB8:3000:3000::42
 domain-name example.com
!
interface Ethernet0/0
 description Access link down to customers
 ipv6 address 2001:DB8:1234:42::1/64
 ipv6 nd other-config-flag
 ipv6 dhcp server dhcp-pool
```

The client has no obvious DHCPv6 configuration. However, the **ipv6 address autoconfig** command on the uplink to the service provider (Ethernet 0/0) causes the following two events:

- Addresses are autoconfigured on the interface, based on prefixes in RA messages received from the server.
- If received RA messages have the "other configuration" flag set, the interface will attempt to acquire the other (for example, nonaddress) configuration from any DHCPv6 servers.

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS master command list, all releases	*Cisco IOS Master Command List, All Releases*
IPv6 commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*

Related Topic	Document Title
IPv6 basic connectivity	"Implementing IPv6 Addressing and Basic Connectivity" module of the *Cisco IOS IPv6 Configuration Guide*
IPv6 prefix delegation	• "Implementing IPv6 Addressing and Basic Connectivity" module of the *Cisco IOS IPv6 Configuration Guide* • "Implementing ADSL and Deploying Dial Access for IPv6" module of the *Cisco IOS IPv6 Configuration Guide*
IPv6 supported feature list	"Start Here: Cisco IOS Software Release Specifics for IPv6 Features" module of the *Cisco IOS IPv6 Configuration Guide*
IPv6 VPN over MPLS	"Implementing IPv6 VPN over MPLS" module of the *Cisco IOS IPv6 Configuration Guide*

Standards and RFCs

Standards/RFCs	Title
RFC 3315	*Dynamic Host Configuration Protocol for IPv6*
RFC 3319	*Dynamic Host Configuration Protocol (DHCPv6) Options for Session Initiation Protocol (SIP) Servers*
RFC 3633	*IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) Version 6*
RFC 3646	*DNS Configuration Options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*
RFC 3736	*Stateless DHCP Service for IPv6*
RFC 3898	*Network Information Service (NIS) Configuration Options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*
RFC 4075	*Simple Network Time Protocol (SNTP) Configuration Option for DHCPv6*
RFC 4242	*Information Refresh Time Option for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*
RFC 4649	*Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Relay Agent Remote-ID Option*
RFC 5460	*DHCPv6 Bulk Leasequery*

MIBs

MIBs	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing DHCP for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 13 Feature Information for Implementing DHCP for IPv6

Feature Name	Releases	Feature Information
DHCP—DHCPv6 Relay Agent Notification for Prefix Delegation	12.2(33)SCA 12.2(33)SRC 12.2(33)SXI 15.0(1)S	DHCPv6 relay agent notification for prefix delegation allows the router working as a DHCPv6 relay agent to find prefix delegation options by reviewing the contents of a DHCPv6 packet that is being relayed by the relay agent to the client.
DHCP—DHCPv6 Server SNTP, NIS, NIS+, Refresh Timer Options	12.4(15)T	The DHCPv6 server options are part of DHCP stateless autoconfiguration.

Feature Name	Releases	Feature Information
DHCPv6 Ethernet Remote ID Option	12.2(33)SRC 12.2(33)SXI 15.0(1)S	This feature adds the remote-ID option to relayed (RELAY-FORWARD) DHCPv6 packets.
IPv6 Access Services—DHCP for IPv6 Relay Agent	12.2(28)SB 12.2(33)SRC 12.2(33)SXI 12.3(11)T 12.4 12.4(2)T	A DHCP relay agent, which may reside on the client's link, is used to relay messages between the client and server.
IPv6 Access Services—DHCPv6 Client Information Refresh Option	12.4(15)T	The DHCPv6 information refresh option can specify an upper boundary for the length of time a client should wait before refreshing information retrieved from DHCPv6.
IPv6 Access Services—DHCPv6 Prefix Delegation	12.0(32)S 12.2(28)SB 12.2(33)SRA 12.2(18)SXE 12.3(4)T 12.4 12.4(2)T 15.0(1)S	The DHCPv6 Prefix Delegation feature can be used to manage link, subnet, and site addressing changes. DHCPv6 can be used in environments to deliver stateful and stateless information.
IPv6 Access Services—DHCPv6 Server Stateless Auto Configuration	12.4(15)T	Hierarchical DHCPv6 for stateless configuration parameters allows a stateless or stateful DHCPv6 client to export configuration parameters (DHCPv6 options) to a local DHCPv6 server pool.
IPv6 Access Services—Stateless DHCPv6	12.2(33)SRA 12.2(18)SXE 12.3(4)T 12.4 12.4(2)T	Stateless DHCPv6 allows DHCPv6 to be used for configuring a node with parameters that do not require a server to maintain any dynamic state for the node.
DHCPv6 Relay—Reload Persistent Interface ID Option	12.2(33)SB 12.2(33)SRC 12.2(33)SXI 15.0(1)S	This feature makes the interface-ID option, which is used by relay agents to decide which interface should be used when forwarding a RELAY-REPLY packet, persistent.
DHCP—DHCPv6 Individual Address Assignment	12.4(24)T	This feature manages nonduplicate address assignment in the correct prefix based on the network where the host is connected.

Feature Name	Releases	Feature Information
DHCP—DHCPv6 Relay SSO/ ISSU	12.2(33)SRE	SSO and ISSU use redundant hardware, with the active and standby RP each running an instance of the DHCP relay agent.
DHCPv6 Bulk—Lease Query	12.2(58)SE 15.1(1)S	Cisco IOS DHCPv6 relay agent supports bulk-lease query in accordance with RFC 5460. The following commands were introduced for this feature: **debug ipv6 dhcp relay**, **ipv6 dhcp-relay bulk-lease**.
DHCPv6—Relay chaining (for Prefix Delegation) and route insertion in FIB	15.2(1)S	This feature allows DHCPv6 messages to be relayed through multiple relay agents. The following commands were introduced or modified by this feature: **clear ipv6 dhcp relay binding**, **clear ipv6 dhcp route**, **ipv6 dhcp iana-route-add** , **ipv6 dhcp iapd-route-add**, **show ipv6 dhcp relay binding**, **show ipv6 dhcp route.**
DHCPv6 Relay Source Configuration	12.2(33)SRE 12.2(58)SE	In some networks that use DHCPv6, it may be desirable to configure a stable address (such as a loopback interface) and have the relay use that interface as the source address of relayed messages. The DHCPv6 relay source configuration feature provides this capability.

Feature Name	Releases	Feature Information
DHCPv6 Repackaging	12.2(33)SRE 12.2(33)XNE	The DHCPv6 repackaging feature consists of DHCPv6 individual address assignment and stateless DHCPv6. The DHCPv6 Individual Address Assignment feature manages nonduplicate address assignment in the correct prefix based on the network where the host is connected. The stateless DHCPv6 feature allows DHCPv6 to be used for configuring a node with parameters that do not require a server to maintain any dynamic state for the node.
DHCPv6 Server—MPLS VPN Support	15.1(2)S	The DHCPv6 server implementation of MPLS VPN support allows a per-pool configuration so DHCPv6 pools can be associated with a VRF instance. The DHCPv6 relay implementation allows the configuration of the destination VRF instance to which the relay messages will be forwarded.
DHCPv6 Server-Relay-Client Support in a VRF Lite Environment	12.2(58)SE	This feature is supported.

Implementing Dynamic Multipoint VPN for IPv6

This document describes how to implement the Dynamic Multipoint VPN for IPv6 feature, which allows users to better scale large and small IPsec Virtual Private Networks (VPNs) by combining generic routing encapsulation (GRE) tunnels, IP security (IPsec) encryption, and the Next Hop Resolution Protocol (NHRP). In Dynamic Multipoint Virtual Private Network (DMVPN) for IPv6, the public network (the Internet) is a pure IPv4 network, and the private network (the intranet) is IPv6 capable.

In Cisco IOS Release 15.2(1)T, IPv6 support on DMVPN was extended to the public network (the Internet) facing the Internet service provider (ISP). The IPv6 transport for DMVPN feature builds IPv6 WAN-side capability into NHRP tunnels and the underlying IPsec encryption, and enables IPv6 to transport payloads on the Internet.

Note The IPv6 transport for DMVPN feature is enabled by default. You need not upgrade your private internal network to IPv6 for the IPv6 transport for DMVPN feature to function. You can have either IPv4 or IPv6 addresses on your local networks.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Implementing DMVPN for IPv6

- This document assumes that you are familiar with IPv6 and IPv4. See the publications referenced in the Additional References, page 259 section for IPv6 and IPv4 configuration and command reference information.
- Perform basic IPv6 addressing and basic connectivity as described in " Implementing IPv6 Addressing and Basic Connectivity ."
- One of the following protocols must be enabled for DMVPN for IPv6 to work: Border Gateway Protocol (BGP), Enhanced Interior Gateway Routing Protocol (EIGRP), On-Demand Routing (ODR), Open Shortest Path First (OSPF), and Routing Information Protocol (RIP).
- Every IPv6 NHRP interface is configured with one IPv6 unicast address. This address can be a globally reachable or unique local address.
- Every IPv6 NHRP interface is configured with one IPv6 link-local address that is unique across all DMVPN hosts in the DMVPN cloud (that is, the hubs and spokes).
- Before a multipoint GRE (mGRE) and IPsec tunnel can be established, you must define an Internet Key Exchange (IKE) policy by using the **crypto isakmp policy** command.

Restrictions for Implementing DMVPN for IPv6

- IPv6 can be configured only on a protected network.
- Every IPv6 NHRP interface is configured with one IPv6 unicast address. This address can be a globally reachable address or a unique local address.
- Every IPv6 NHRP interface is configured with one IPv6 link-local address that is unique across all DMVPN nodes in the DMVPN cloud (that is, the hubs and spokes).
- IPv6 VRFs are not fully supported by IPv6 routing protocols such as EIGRP or OSPF. Therefore, DMVPN for IPv6 does not support IPv6 VRFs.
- Per tunnel QoS, DHCP-Tunnels Support, and 2547oDMVPN--Enabling Traffic Segmentation within DMVPN features are not supported for IPv6.
- Internet Key Exchange version 1 (IKEv1) and Network Address Translation 66 (NAT66) are not supported.

Information About Implementing DMVPN for IPv6

DMVPN for IPv6 Overview

The DMVPN feature combines NHRP routing, multipoint generic routing encapsulation (mGRE) tunnels, and IPsec encryption to provide users ease of configuration via crypto profiles--which override the requirement for defining static crypto maps--and dynamic discovery of tunnel endpoints.

This feature relies on the following Cisco enhanced standard technologies:

- NHRP--A client and server protocol where the hub is the server and the spokes are the clients. The hub maintains an NHRP database of the public interface addresses of each spoke. Each spoke registers its

real address when it boots and queries the NHRP database for real addresses of the destination spokes to build direct tunnels.

- mGRE tunnel interface--An mGRE tunnel interface allows a single GRE interface to support multiple IPsec tunnels and simplifies the size and complexity of the configuration.
- IPsec encryption--An IPsec tunnel interface facilitates for the protection of site-to-site IPv6 traffic with native encapsulation.

In DMVPN for IPv6, the public network (the Internet) is a pure IPv4 network, and the private network (the intranet) is IPv6 capable. The intranets could be a mix of IPv4 or IPv6 clouds connected to each other using DMVPN technologies, with the underlying carrier being a traditional IPv4 network.

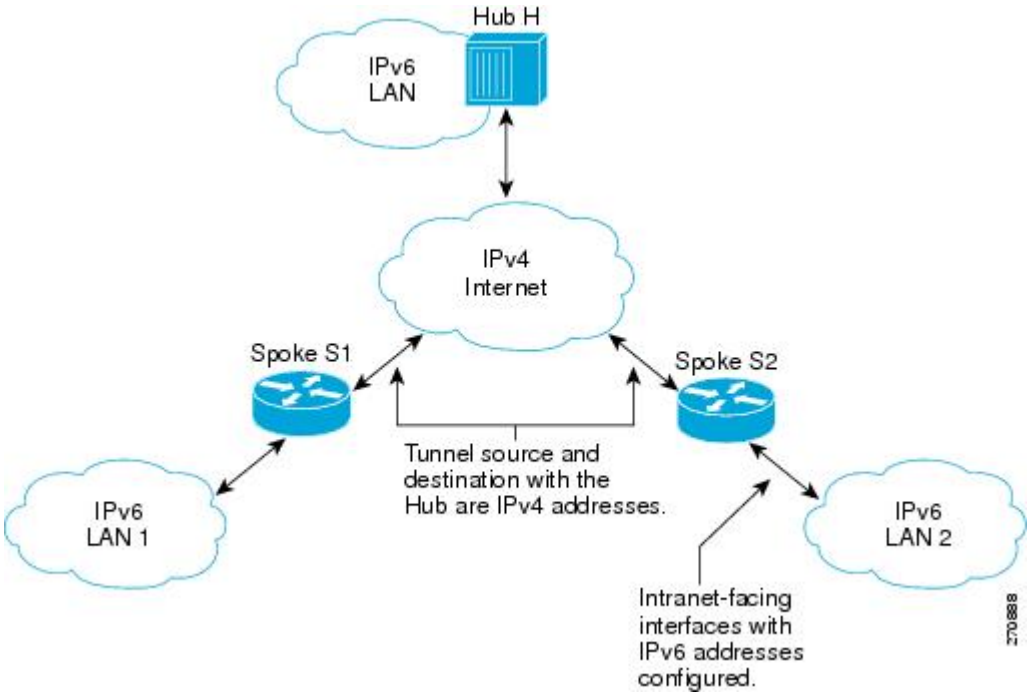
- NHRP Routing, page 237
- IPv6 Routing, page 238
- IPv6 Addressing and Restrictions, page 238

NHRP Routing

The NHRP protocol resolves a given intranet address (IPv4 or IPv6) to an Internet address (IPv4 nonbroadcast multiaccess [NBMA] address).

In the figure below, the intranets that are connected over the DMVPN network are IPv6 clouds, and the Internet is a pure IPv4 cloud. Spokes S1 and S2 are connected to Hub H over the Internet using a statically configured tunnel. The address of the tunnel itself is the IPv6 domain, because it is another node on the intranet. The source and destinations address of the tunnel (the mGRE endpoints), however, are always in IPv4, in the Internet domain. The mGRE tunnel is aware of the IPv6 network because the GRE passenger protocol is an IPv6 packet, and the GRE transport (or carrier) protocol is an IPv4 packet.

Figure 22 IPv6 Topology That Triggers NHRP



When an IPv6 host in LAN L1 sends a packet destined to an IPv6 host in LAN L2, the packet is first routed to the gateway (which is Spoke S1) in LAN L1. Spoke S1 is a dual-stack router, which means both IPv4

and IPv6 are configured on it. The IPv6 routing table in S1 points to a next hop, which is the IPv6 address of the tunnel on Spoke S2. This is a VPN address that must be mapped to an NBMA address, triggering NHRP.

- IPv6 NHRP Redirect and Shortcut Features, page 238

IPv6 NHRP Redirect and Shortcut Features

When IPv6 NHRP redirect is enabled, NHRP examines every data packet in the output feature path. If the data packet enters and leaves on the same logical network, NHRP sends an NHRP traffic indication message to the source of the data packet. In NHRP, a logical network is identified by the NHRP network ID, which groups multiple physical interfaces into a single logical network.

When IPv6 NHRP shortcut is enabled, NHRP intercepts every data packet in the output feature path. It checks to see if there is an NHRP cache entry to the destination of the data packet and, if yes, it replaces the current output adjacency with the one present in the NHRP cache. The data packet is therefore switched out using the new adjacency provided by NHRP.

IPv6 Routing

NHRP is automatically invoked for mGRE tunnels carrying the IPv6 passenger protocol. When a packet is routed and sent to the switching path, NHRP looks up the given next hop and, if required, initiates an NHRP resolution query. If the resolution is successful, NHRP populates the tunnel endpoint database, which in turn populates the Cisco Express Forwarding adjacency table. The subsequent packets are Cisco Express Forwarding switched if Cisco Express Forwarding is enabled.

IPv6 Addressing and Restrictions

IPv6 allows multiple unicast addresses on a given IPv6 interface. IPv6 also allows special address types, such as anycast, multicast, link-local addresses, and unicast addresses.

DMVPN for IPv6 has the following addressing restrictions:

- Every IPv6 NHRP interface is configured with one IPv6 unicast address. This address can be a globally reachable or unique local address.
- Every IPv6 NHRP interface is configured with one IPv6 link-local address that is unique across all DMVPN hosts in the DMVPN cloud (that is, the hubs and spokes).
  - If no other tunnels on the router are using the same tunnel source, then the tunnel source address can be embedded into an IPv6 address.
  - If the router has only one DMVPN IPv6 tunnel, then manual configuration of the IPv6 link-local address is not required. Instead, use the **ipv6 enable** command to autogenerate a link-local address.
  - If the router has more than one DMVPN IPv6 tunnel, then the link-local address must be manually configured using the **ipv6 address fe80::2001 link-local** command.

mGRE Support over IPv6

Multiple sites of a DMVPN are interconnected by IPv6. A single logical mGRE tunnel interface interconnects one VPN site to another. An IPv6 subnet connects a tunnel interface with other tunnel interfaces from various VPN sites. All tunnel interfaces connecting VPN sites act as hosts on the logical IPv6 subnet. This structure is referred to as the tunnel overlay network.

How to Configure DMVPN for IPv6

To enable mGRE and IPsec tunneling for hub and spoke routers, you must configure an IPsec profile that uses a global IPsec policy template and configure your mGRE tunnel for IPsec encryption. This section contains the following procedures:

Configuring an IPsec Profile in DMVPN for IPv6

The IPsec profile shares most commands with the crypto map configuration, but only a subset of the commands are valid in an IPsec profile. Only commands that pertain to an IPsec policy can be issued under an IPsec profile; you cannot specify the IPsec peer address or the access control list (ACL) to match the packets that are to be encrypted.

Before configuring an IPsec profile, you must do the following:

- Define a transform set by using the **crypto ipsec transform-set** command.
- Make sure that the Internet Security Association Key Management Protocol (ISAKMP) profile is configured with default ISAKMP settings.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto identity** *name*
4. **exit**
5. **crypto ipsec profile** *name*
6. **set transform-set** *transform-set-name*
7. **set identity**
8. **set security-association lifetime seconds** *seconds* | **kilobytes** *kilobytes*
9. **set pfs** [**group1** | **group2**
10. end

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto identity** *name* **Example:** `Router(config)# crypto identity router1`	Configures the identity of the router with a given list of distinguished names (DNs) in the certificate of the router.
Step 4	**exit** **Example:** `Router(config-crypto-identity)# exit`	Exits crypto identity configuration mode and enters global configuration mode.
Step 5	**crypto ipsec profile** *name* **Example:** `Router(config)# crypto ipsec profile example1`	Defines the IPsec parameters that are to be used for IPsec encryption between "spoke and hub" and "spoke and spoke" routers. This command places the router in crypto map configuration mode.
Step 6	**set transform-set** *transform-set-name* **Example:** `Router(config-crypto-map)# set transform-set example-set`	Specifies which transform sets can be used with the IPsec profile.
Step 7	**set identity** **Example:** `Router(config-crypto-map)# set identity router1`	(Optional) Specifies identity restrictions to be used with the IPsec profile.

	Command or Action	Purpose
Step 8	**set security-association lifetime seconds** *seconds* | **kilobytes** *kilobytes* **Example:** `Router(config-crypto-map)# set security-association lifetime seconds 1800`	(Optional) Overrides the global lifetime value for the IPsec profile.
Step 9	**set pfs** [**group1** | **group2** **Example:** `Router(config-crypto-map)# set pfs group2`	(Optional) Specifies that IPsec should ask for perfect forward secrecy (PFS) when requesting new security associations for this IPsec profile.
Step 10	end **Example:** `Router(config-crypto-map)# end`	Exits crypto map configuration mode and returns to privileged EXEC mode.

Configuring the Hub for IPv6 over DMVPN

Perform this task to configure the hub router for IPv6 over DMVPN for mGRE and IPsec integration (that is, associate the tunnel with the IPsec profile configured in the previous procedure).

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *number*
4. **ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits* / *prefix-length*
5. **ipv6 address** *ipv6-address* / *prefix-length* **link-local**
6. **ipv6 mtu** *bytes*
7. **ipv6 nhrp authentication** *string*
8. **ipv6 nhrp map multicast dynamic**
9. **ipv6 nhrp network-id** *network-id*
10. **tunnel source** *ip-address* | *ipv6-address* | *interface-type interface-number*
11. **tunnel mode** {**aurp** | **cayman** | **dvmrp** | **eon** | **gre**| **gre multipoint**[**ipv6**] | **gre ipv6** | **ipip decapsulate-any**] | **ipsec ipv4** | **iptalk** | **ipv6**| **ipsec ipv6** | **mpls** | **nos** | **rbscp**
12. **tunnel protection ipsec profile** *name* [**shared**]
13. **bandwidth** {*kbps* | **inherit** [*kbps*] | **receive** [*kbps*]}
14. **ipv6 nhrp holdtime** *seconds*
15. end

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface tunnel** *number* **Example:** `Router(config)# interface tunnel 5`	Configures a tunnel interface and enters interface configuration mode. • The number argument specifies the number of the tunnel interfaces that you want to create or configure. There is no limit on the number of tunnel interfaces you can create.
Step 4	**ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits* / *prefix-length* **Example:** `Router(config-if)# ipv6 address 2001:DB8:1:1::72/64`	Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.
Step 5	**ipv6 address** *ipv6-address* / *prefix-length* **link-local** **Example:** `Router(config-if)# ipv6 address fe80::2001 link-local`	Configures an IPv6 link-local address for an interface and enables IPv6 processing on the interface. • A unique IPv6 link-local address (across all DMVPN nodes in a DMVPN network) must be configured.
Step 6	**ipv6 mtu** *bytes* **Example:** `Router(config-if)# ipv6 mtu 1400`	Sets the maximum transmission unit (MTU) size of IPv6 packets sent on an interface.

	Command or Action	Purpose
Step 7	**ipv6 nhrp authentication** *string* **Example:** `Router(config-if)# ipv6 nhrp authentication examplexx`	Configures the authentication string for an interface using the NHRP. **Note** The NHRP authentication string must be set to the same value on all hubs and spokes that are in the same DMVPN network.
Step 8	**ipv6 nhrp map multicast dynamic** **Example:** `Router(config-if)# ipv6 nhrp map multicast dynamic`	Allows NHRP to automatically add routers to the multicast NHRP mappings.
Step 9	**ipv6 nhrp network-id** *network-id* **Example:** `Router(config-if)# ipv6 nhrp network-id 99`	Enables the NHRP on an interface.
Step 10	**tunnel source** *ip-address* | *ipv6-address* | *interface-type interface-number* **Example:** `Router(config-if)# tunnel source ethernet 0`	Sets the source address for a tunnel interface.
Step 11	**tunnel mode** {**aurp** | **cayman** | **dvmrp** | **eon** | **gre**| **gre multipoint**[**ipv6**] | **gre ipv6** | **ipip decapsulate-any**] | **ipsec ipv4** | **iptalk** | **ipv6**| **ipsec ipv6** | **mpls** | **nos** | **rbscp** **Example:** `Router(config-if)# tunnel mode gre multipoint`	Sets the encapsulation mode to mGRE for the tunnel interface.
Step 12	**tunnel protection ipsec profile** *name* [**shared**] **Example:** `Router(config-if)# tunnel protection ipsec profile example_profile`	Associates a tunnel interface with an IPsec profile. • The name argument specifies the name of the IPsec profile; this value must match the name specified in the **crypto ipsec profile name** command.

	Command or Action	Purpose
Step 13	**bandwidth** {*kbps* | **inherit** [*kbps*] | **receive** [*kbps*]} **Example:** `Router(config-if)# bandwidth 1200`	Sets the current bandwidth value for an interface to higher-level protocols. • The *bandwidth-size* argument specifies the bandwidth in kilobits per second. The default value is 9. The recommended bandwidth value is 1000 or greater.
Step 14	**ipv6 nhrp holdtime** *seconds* **Example:** `Router(config-if)# ipv6 nhrp holdtime 3600`	Changes the number of seconds that NHRP NBMA addresses are advertised as valid in authoritative NHRP responses.
Step 15	end **Example:** `Router(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring the NHRP Redirect and Shortcut Features on the Hub

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel number**
4. **ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits* / *prefix-length*
5. **ipv6 nhrp redirect timeout** *seconds*]
6. **ipv6 nhrp shortcut**
7. end

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface tunnel number** **Example:** `Router(config)# interface tunnel 5`	Configures a tunnel interface and enters interface configuration mode. • The number argument specifies the number of the tunnel interfaces that you want to create or configure. There is no limit on the number of tunnel interfaces you can create.
Step 4	**ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits* / *prefix-length* **Example:** `Router(config-if)# ipv6 address 2001:DB8:1:1::72/64`	Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.
Step 5	**ipv6 nhrp redirect timeout** *seconds*] **Example:** `Router(config-if)# ipv6 nhrp redirect`	Enables NHRP redirect. **Note** You must configure the **ipv6 nhrp redirect** command on a hub.
Step 6	**ipv6 nhrp shortcut** **Example:** `Router(config-if)# ipv6 nhrp shortcut`	Enables NHRP shortcut switching. **Note** You must configure the **ipv6 nhrp shortcut** command on a spoke.
Step 7	end **Example:** `Router(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring the Spoke for IPv6 over DMVPN

Perform this task to configure the spoke for IPv6 over DMVPN.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *number*
4. **ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits* / *prefix-length*
5. **ipv6 address** *ipv6-address* / *prefix-length* **link-local**
6. **ipv6 mtu** *bytes*
7. **ipv6 nhrp authentication** *string*
8. **ipv6 nhrp map** *ipv6-address nbma-address*
9. **ipv6 nhrp map multicast** *ipv4-nbma-address*
10. **ipv6 nhrp nhs** *ipv6- nhs-address*
11. **ipv6 nhrp network-id** *network-id*
12. **tunnel source** *ip-address* | *ipv6-address* | *interface-type interface-number*
13. Do one of the following:
    - **tunnel mode** {**aurp** | **cayman** | **dvmrp** | **eon** | **gre**| **gre multipoint** [**ipv6**] | **gre ipv6** | **ipip decapsulate-any**] | **ipsec ipv4** | **iptalk** | **ipv6**| **ipsec ipv6** | **mpls** | **nos** | **rbscp**
    -
    -
    - **tunnel destination** {*host-name* | *ip-address* | *ipv6-address*}
14. **tunnel protection ipsec profile** *name* [**shared**]
15. **bandwidth** {**interzone** | **total** | **session**} {**default** | **zone** *zone-name*} *bandwidth-size*
16. **ipv6 nhrp holdtime** *seconds*
17. end

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface tunnel** *number* **Example:** `Router(config)# interface tunnel 5`	Configures a tunnel interface and enters interface configuration mode. • The *number* argument specifies the number of the tunnel interfaces that you want to create or configure. There is no limit on the number of tunnel interfaces you can create.
Step 4	**ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits* / *prefix-length* **Example:** `Router(config-if) ipv6 address 2001:DB8:1:1::72/64`	Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.
Step 5	**ipv6 address** *ipv6-address* / *prefix-length* **link-local** **Example:** `Router(config-if)# ipv6 address fe80::2001 link-local`	Configures an IPv6 link-local address for an interface and enables IPv6 processing on the interface. • A unique IPv6 link-local address (across all DMVPN nodes in a DMVPN network) must be configured.
Step 6	**ipv6 mtu** *bytes* **Example:** `Router(config-if)# ipv6 mtu 1400`	Sets the MTU size of IPv6 packets sent on an interface.
Step 7	**ipv6 nhrp authentication** *string* **Example:** `Router(config-if)# ipv6 nhrp authentication examplexx`	Configures the authentication string for an interface using the NHRP. **Note** The NHRP authentication string must be set to the same value on all hubs and spokes that are in the same DMVPN network.
Step 8	**ipv6 nhrp map** *ipv6-address nbma-address* **Example:** `Router(config-if)# ipv6 nhrp map 2001:DB8:3333:4::5 10.1.1.1`	Statically configures the IPv6-to-NBMA address mapping of IPv6 destinations connected to an NBMA network. **Note** Only IPv4 NBMA addresses are supported, not ATM or Ethernet addresses.

	Command or Action	Purpose
Step 9	**ipv6 nhrp map multicast** *ipv4-nbma-address* **Example:** `Router(config-if)# ipv6 nhrp map multicast 10.11.11.99`	Maps destination IPv6 addresses to IPv4 NBMA addresses.
Step 10	**ipv6 nhrp nhs** *ipv6- nhs-address* **Example:** `Router(config-if)# ipv6 nhrp nhs 2001:0DB8:3333:4::5 2001:0DB8::/64`	Specifies the address of one or more IPv6 NHRP servers.
Step 11	**ipv6 nhrp network-id** *network-id* **Example:** `Router(config-if)# ipv6 nhrp network-id 99`	Enables the NHRP on an interface.
Step 12	**tunnel source** *ip-address* | *ipv6-address* | *interface-type interface-number* **Example:** `Router(config-if)# tunnel source ethernet 0`	Sets the source address for a tunnel interface.

Command or Action	Purpose

Step 13 Do one of the following:
• **tunnel mode** {**aurp** | **cayman** | **dvmrp** | **eon** | **gre**| **gre multipoint** [**ipv6**] | **gre ipv6** | **ipip decapsulate-any**] | **ipsec ipv4** | **iptalk** | **ipv6**| **ipsec ipv6** | **mpls** | **nos** | **rbscp**
•
•
• **tunnel destination** {*host-name* | *ip-address* | *ipv6-address*}
Example:
`Router(config-if)# tunnel mode gre multipoint`
Example:
Example:
Example:
`Router(config-if)# tunnel destination 10.1.1.1`
	Sets the encapsulation mode to mGRE for the tunnel interface.
	• Use the **tunnel mode** command if data traffic can use dynamic spoke-to-spoke traffic.
	or
	Specifies the destination for a tunnel interface.
	• Use the **tunnel destination** command if data traffic can use hub-and-spoke tunnels.

Step 14 **tunnel protection ipsec profile** *name* [**shared**] **Example:** `Router(config-if)# tunnel protection ipsec profile example1`	Associates a tunnel interface with an IPsec profile. • The *name* argument specifies the name of the IPsec profile; this value must match the name specified in the **crypto ipsec profile name** command.
Step 15 **bandwidth** {**interzone** | **total** | **session**} {**default** | **zone** *zone-name*} *bandwidth-size* **Example:** `Router(config-if)# bandwidth total 1200`	Sets the current bandwidth value for an interface to higher-level protocols. • The *bandwidth-size* argument specifies the bandwidth in kilobits per second. The default value is 9. The recommended bandwidth value is 1000 or greater. • The bandwidth setting for the spoke need not equal the bandwidth setting for the DMVPN hub. It is usually easier if all of the spokes use the same or similar value.

	Command or Action	Purpose
Step 16	**ipv6 nhrp holdtime** *seconds* **Example:** `Router(config-if)# ipv6 nhrp holdtime 3600`	Changes the number of seconds that NHRP NBMA addresses are advertised as valid in authoritative NHRP responses.
Step 17	end **Example:** `Router(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Verifying DMVPN for IPv6 Configuration

SUMMARY STEPS

1. **enable**
2. **show dmvpn** [**ipv4** [**vrf** *vrf-name*] | **ipv6** [**vrf** *vrf-name*]] [**debug-condition** | [**interface tunnel** *number* | **peer** {**nbma** *ip-address* | **network** *network-mask* | **tunnel** *ip-address*}] [**static**] [**detail**]]
3. **show ipv6 nhrp** [**dynamic** [*ipv6-address*] | **incomplete** | **static**] [*address* | *interface*] [**brief** | **detail**] [**purge**]
4. **show ipv6 nhrp multicast** [*ipv4-address* | *interface* | *ipv6-address*]
5. **show ip nhrp multicast** [*nbma-address* | *interface*]
6. **show ipv6 nhrp summary**
7. **show ipv6 nhrp traffic** [*interface***tunnel** *number*
8. **show ip nhrp shortcut**
9. **show ip route**
10. **show ipv6 route**
11. **show nhrp debug-condition**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**show dmvpn** [**ipv4** [**vrf** *vrf-name*] | **ipv6** [**vrf** *vrf-name*]] [**debug-condition** | [**interface tunnel** *number* | **peer** {**nbma** *ip-address* | **network** *network-mask* | **tunnel** *ip-address*}] [**static**] [**detail**]] **Example:** `Router# show dmvpn 2001:0db8:1:1::72/64`	Displays DMVPN-specific session information.
Step 3	**show ipv6 nhrp** [**dynamic** [*ipv6-address*] | **incomplete** | **static**] [*address* | *interface*] [**brief** | **detail**] [**purge**] **Example:** `Router# show ipv6 nhrp`	Displays NHRP mapping information.
Step 4	**show ipv6 nhrp multicast** [*ipv4-address* | *interface* | *ipv6-address*] **Example:** `Router# show ipv6 nhrp multicast`	Displays NHRP multicast mapping information.
Step 5	**show ip nhrp multicast** [*nbma-address* | *interface*] **Example:** `Router#` **`show ip nhrp multicast`**	Displays NHRP multicast mapping information.
Step 6	**show ipv6 nhrp summary** **Example:** `Router# show ipv6 nhrp summary`	Displays NHRP mapping summary information.
Step 7	**show ipv6 nhrp traffic** [*interface***tunnel** *number* **Example:** `Router# show ipv6 nhrp traffic`	Displays NHRP traffic statistics information.
Step 8	**show ip nhrp shortcut** **Example:** `Router# show ip nhrp shortcut`	Displays NHRP shortcut information.

	Command or Action	Purpose
Step 9	**show ip route** **Example:** `Router# show ip route`	Displays the current state of the IPv4 routing table.
Step 10	**show ipv6 route** **Example:** `Router# show ipv6 route`	Displays the current contents of the IPv6 routing table.
Step 11	**show nhrp debug-condition** **Example:** `Router# show nhrp debug-condition`	Displays the NHRP conditional debugging information.

Examples

Sample Output from the show dmvpn Command

The following sample output is from the **show dmvpn**command, with the **ipv6** and **detail** keywords, for the hub:

```
Router# show dmvpn ipv6 detail
Legend: Attrb --> S - Static, D - Dynamic, I - Incomplete
        N - NATed, L - Local, X - No Socket
        # Ent --> Number of NHRP entries with same NBMA peer
        NHS Status: E --> Expecting Replies, R --> Responding
        UpDn Time --> Up or Down Time for a Tunnel
==========================================================================
Interface Tunnel1 is up/up, Addr. is 10.0.0.3, VRF ""
   Tunnel Src./Dest. addr: 192.169.2.9/MGRE, Tunnel VRF ""
   Protocol/Transport: "multi-GRE/IP", Protect "test_profile"
Type:Hub, Total NBMA Peers (v4/v6): 2
    1.Peer NBMA Address: 192.169.2.10
        Tunnel IPv6 Address: 2001::4
        IPv6 Target Network: 2001::4/128
        # Ent: 2, Status: UP, UpDn Time: 00:01:51, Cache Attrib: D
Type:Hub, Total NBMA Peers (v4/v6): 2
    2.Peer NBMA Address: 192.169.2.10
        Tunnel IPv6 Address: 2001::4
        IPv6 Target Network: FE80::2/128
        # Ent: 0, Status: UP, UpDn Time: 00:01:51, Cache Attrib: D
Type:Hub, Total NBMA Peers (v4/v6): 2
    3.Peer NBMA Address: 192.169.2.11
        Tunnel IPv6 Address: 2001::5
        IPv6 Target Network: 2001::5/128
        # Ent: 2, Status: UP, UpDn Time: 00:26:38, Cache Attrib: D
Type:Hub, Total NBMA Peers (v4/v6): 2
    4.Peer NBMA Address: 192.169.2.11
        Tunnel IPv6 Address: 2001::5
        IPv6 Target Network: FE80::3/128
        # Ent: 0, Status: UP, UpDn Time: 00:26:38, Cache Attrib: D
```

```
Pending DMVPN Sessions:
Interface: Tunnel1
  IKE SA: local 192.169.2.9/500 remote 192.169.2.10/500 Active
  Crypto Session Status: UP-ACTIVE
  fvrf: (none), Phase1_id: 192.169.2.10
  IPSEC FLOW: permit 47 host 192.169.2.9 host 192.169.2.10
        Active SAs: 2, origin: crypto map
   Outbound SPI : 0x BB0ED02, transform : esp-3des esp-sha-hmac
    Socket State: Open
Interface: Tunnel1
  IKE SA: local 192.169.2.9/500 remote 192.169.2.11/500 Active
  Crypto Session Status: UP-ACTIVE
  fvrf: (none), Phase1_id: 192.169.2.11
  IPSEC FLOW: permit 47 host 192.169.2.9 host 192.169.2.11
        Active SAs: 2, origin: crypto map
   Outbound SPI : 0xB79B277B, transform : esp-3des esp-sha-hmac
    Socket State: Open
```

The following sample output is from the **show dmvpn**command, with the **ipv6** and **detail** keywords, for the spoke:

```
Router# show dmvpn ipv6 detail

Legend: Attrb --> S - Static, D - Dynamic, I - Incomplete
        N - NATed, L - Local, X - No Socket
        # Ent --> Number of NHRP entries with same NBMA peer
        NHS Status: E --> Expecting Replies, R --> Responding
        UpDn Time --> Up or Down Time for a Tunnel
==========================================================================
Interface Tunnel1 is up/up, Addr. is 10.0.0.1, VRF ""
   Tunnel Src./Dest. addr: 192.169.2.10/MGRE, Tunnel VRF ""
   Protocol/Transport: "multi-GRE/IP", Protect "test_profile"
IPv6 NHS: 2001::6 RE
Type:Spoke, Total NBMA Peers (v4/v6): 1
    1.Peer NBMA Address: 192.169.2.9
        Tunnel IPv6 Address: 2001::6
        IPv6 Target Network: 2001::/112
        # Ent: 2, Status: NHRP, UpDn Time: never, Cache Attrib: S
IPv6 NHS: 2001::6 RE
Type:Unknown, Total NBMA Peers (v4/v6): 1
    2.Peer NBMA Address: 192.169.2.9
        Tunnel IPv6 Address: FE80::1
        IPv6 Target Network: FE80::1/128
        # Ent: 0, Status: UP, UpDn Time: 00:00:24, Cache Attrib: D
Pending DMVPN Sessions:
Interface: Tunnel1
  IKE SA: local 192.169.2.10/500 remote 192.169.2.9/500 Active
  Crypto Session Status: UP-ACTIVE
  fvrf: (none), Phase1_id: 192.169.2.9
  IPSEC FLOW: permit 47 host 192.169.2.10 host 192.169.2.9
        Active SAs: 2, origin: crypto map
   Outbound SPI : 0x6F75C431, transform : esp-3des esp-sha-hmac
    Socket State: Open
```

Sample Output from the show ipv6 nhrp Command

The following sample output is from the **show ipv6 nhrp**command for the hub and the spoke:

Hub

```
Router# show ipv6 nhrp
2001::4/128 via 2001::4
   Tunnel1 created 00:02:40, expire 00:00:47
   Type: dynamic, Flags: unique registered used
   NBMA address: 192.169.2.10
2001::5/128 via 2001::5
   Tunnel1 created 00:02:37, expire 00:00:47
   Type: dynamic, Flags: unique registered used
   NBMA address: 192.169.2.11
FE80::2/128 via 2001::4
```

```
   Tunnel1 created 00:02:40, expire 00:00:47
   Type: dynamic, Flags: unique registered used
   NBMA address: 192.169.2.10
FE80::3/128 via 2001::5
   Tunnel1 created 00:02:37, expire 00:00:47
   Type: dynamic, Flags: unique registered used
   NBMA address: 192.169.2.11
```

Spoke

```
Router# show ipv6 nhrp
2001::8/128
   Tunnel1 created 00:00:13, expire 00:02:51
   Type: incomplete, Flags: negative
   Cache hits: 2
2001::/112 via 2001::6
   Tunnel1 created 00:01:16, never expire
   Type: static, Flags: used
   NBMA address: 192.169.2.9
FE80::1/128 via FE80::1
   Tunnel1 created 00:01:15, expire 00:00:43
   Type: dynamic, Flags:
   NBMA address: 192.169.2.9
```

Sample Output from the show ipv6 nhrp multicast Command

The following sample output is from the **show ipv6 nhrp multicast**command for the hub and the spoke:

Hub

```
Router# show ipv6 nhrp multicast
  I/F     NBMA address
Tunnel1    192.169.2.10    Flags: dynamic
Tunnel1    192.169.2.11    Flags: dynamic
```

Spoke

```
Router# show ipv6 nhrp multicast
  I/F     NBMA address
Tunnel1    192.169.2.9     Flags: static
```

Sample Output for the show ipv6 nhrp traffic Command

The following sample output is from the **show ipv6 nhrp traffic**command:

```
Router# show ipv6 nhrp traffic
Tunnel0: Max-send limit:100Pkts/10Sec, Usage:0%
Sent: Total 8
1 Resolution Request 1 Resolution Reply 6 Registration Request
0 Registration Reply 0 Purge Request 0 Purge Reply
0 Error Indication 0 Traffic Indication
Rcvd: Total 5
1 Resolution Request 1 Resolution Reply 0 Registration Request
2 Registration Reply 0 Purge Request 0 Purge Reply
0 Error Indication 1 Traffic Indication
```

Monitoring and Maintaining DMVPN for IPv6 Configuration and Operation

SUMMARY STEPS

1. **enable**
2. **clear dmvpn session** [**interface tunnel** *number* | **peer** {*ipv4-address* | *fqdn-string* | *ipv6-address*} | **vrf** *vrf-name*] [**static**]
3. **clear ipv6 nhrp** [*ipv6-address* | **counters**
4. **debug dmvpn** {**all** | **error** | **detail** | **packet**} {**all** | *debug-type*}
5. **debug nhrp** [**cache** | **extension** | **packet** | **rate**]
6. **debug nhrp condition** [**interface tunnel** *number* | **peer** {**nbma** {*ipv4-address* | *fqdn-string* | *ipv6-address*} | **tunnel** {*ip-address* | *ipv6-address*}} | **vrf** *vrf-name*]
7. **debug nhrp error**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear dmvpn session** [**interface tunnel** *number* | **peer** {*ipv4-address* | *fqdn-string* | *ipv6-address*} | **vrf** *vrf-name*] [**static**] **Example:** `Device# clear dmvpn session`	Clears DMVPN sessions.
Step 3	**clear ipv6 nhrp** [*ipv6-address* | **counters** **Example:** `Device# clear ipv6 nhrp`	Clears all dynamic entries from the NHRP cache.
Step 4	**debug dmvpn** {**all** | **error** | **detail** | **packet**} {**all** | *debug-type*} **Example:** `Device# debug dmvpn`	Displays debug DMVPN session information.

	Command or Action	Purpose
Step 5	**debug nhrp** [**cache** | **extension** | **packet** | **rate**] **Example:** `Device# debug nhrp ipv6`	Enables NHRP debugging.
Step 6	**debug nhrp condition** [**interface tunnel** *number* | **peer** {**nbma** {*ipv4-address* | *fqdn-string* | *ipv6-address*} | **tunnel** {*ip-address* | *ipv6-address*}} | **vrf** *vrf-name*] **Example:** `Device# debug nhrp condition`	Enables NHRP conditional debugging.
Step 7	**debug nhrp error** **Example:** `Device# debug nhrp ipv6 error`	Displays NHRP error-level debugging information.

Examples

Sample Output for the debug nhrp Command

The following sample output is from the **debug nhrp**command with the **ipv6** keyword:

```
Device# debug nhrp ipv6
Aug  9 13:13:41.486: NHRP: Attempting to send packet via DEST
            - 2001:DB8:3c4d:0015:0000:0000:1a2f:3d2c/32
Aug  9 13:13:41.486: NHRP: Encapsulation succeeded.
Aug  9 13:13:41.486: NHRP: Tunnel NBMA addr 11.11.11.99
Aug  9 13:13:41.486: NHRP: Send Registration Request via Tunnel0 vrf 0, packet size: 105
Aug  9 13:13:41.486: src: 2001:DB8:3c4d:0015:0000:0000:1a2f:3d2c/32,
             dst: 2001:DB8:3c4d:0015:0000:0000:1a2f:3d2c/32
Aug  9 13:13:41.486: NHRP: 105 bytes out Tunnel0
Aug  9 13:13:41.486: NHRP: Receive Registration Reply via Tunnel0 vrf 0, packet size: 125
```

Configuration Examples for Implementing DMVPN for IPv6

Example: Configuring an IPsec Profile

```
Router(config)# crypto identity router1

Router(config)# crypto ipsec profile example1
```

```
Router(config-crypto-map)# set transform-set example-set
Router(config-crypto-map)# set identity router1

Router(config-crypto-map)# set security-association lifetime seconds 1800

Router(config-crypto-map)# set pfs group2
```

Example: Configuring the Hub for DMVPN

```
Router# configure terminal
Router(config)# interface tunnel 5

Router(config-if)# ipv6 address 2001:DB8:1:1::72/64
Router(config-if)# ipv6 address fe80::2001 link-local
Router(config-if)# ipv6 mtu 1400
Router(config-if)# ipv6 nhrp authentication examplexx
Router(config-if)# ipv6 nhrp map multicast dynamic
Router(config-if)# ipv6 nhrp network-id 99
Router(config-if)# tunnel source ethernet 0
Router(config-if)# tunnel mode gre multipoint
Router(config-if)# tunnel protection ipsec profile example_profile
Router(config-if)# bandwidth 1200
Router(config-if)# ipv6 nhrp holdtime 3600
```

The following sample output is from the **show dmvpn** command, with the **ipv6** and **detail** keywords, for the hub:

```
Router# show dmvpn ipv6 detail

Legend: Attrb --> S - Static, D - Dynamic, I - Incomplete
        N - NATed, L - Local, X - No Socket
        # Ent --> Number of NHRP entries with same NBMA peer
        NHS Status: E --> Expecting Replies, R --> Responding
        UpDn Time --> Up or Down Time for a Tunnel
==========================================================================

Interface Tunnel1 is up/up, Addr. is 10.0.0.3, VRF ""
   Tunnel Src./Dest. addr: 192.169.2.9/MGRE, Tunnel VRF ""
   Protocol/Transport: "multi-GRE/IP", Protect "test_profile"
Type:Hub, Total NBMA Peers (v4/v6): 2
    1.Peer NBMA Address: 192.169.2.10
        Tunnel IPv6 Address: 2001::4
        IPv6 Target Network: 2001::4/128
        # Ent: 2, Status: UP, UpDn Time: 00:01:51, Cache Attrib: D
Type:Hub, Total NBMA Peers (v4/v6): 2
    2.Peer NBMA Address: 192.169.2.10
        Tunnel IPv6 Address: 2001::4
        IPv6 Target Network: FE80::2/128
        # Ent: 0, Status: UP, UpDn Time: 00:01:51, Cache Attrib: D
Type:Hub, Total NBMA Peers (v4/v6): 2
    3.Peer NBMA Address: 192.169.2.11
Tunnel IPv6 Address: 2001::5
        IPv6 Target Network: 2001::5/128
        # Ent: 2, Status: UP, UpDn Time: 00:26:38, Cache Attrib: D
Type:Hub, Total NBMA Peers (v4/v6): 2
    4.Peer NBMA Address: 192.169.2.11
        Tunnel IPv6 Address: 2001::5
        IPv6 Target Network: FE80::3/128
        # Ent: 0, Status: UP, UpDn Time: 00:26:38, Cache Attrib: D
Pending DMVPN Sessions:

Interface: Tunnel1
  IKE SA: local 192.169.2.9/500 remote 192.169.2.10/500 Active
  Crypto Session Status: UP-ACTIVE
  fvrf: (none), Phase1_id: 192.169.2.10
  IPSEC FLOW: permit 47 host 192.169.2.9 host 192.169.2.10
        Active SAs: 2, origin: crypto map
   Outbound SPI : 0x BB0ED02, transform : esp-3des esp-sha-hmac
```

```
    Socket State: Open

Interface: Tunnel1
  IKE SA: local 192.169.2.9/500 remote 192.169.2.11/500 Active
  Crypto Session Status: UP-ACTIVE
  fvrf: (none), Phase1_id: 192.169.2.11
  IPSEC FLOW: permit 47 host 192.169.2.9 host 192.169.2.11
        Active SAs: 2, origin: crypto map
   Outbound SPI : 0xB79B277B, transform : esp-3des esp-sha-hmac
    Socket State: Open
```

Example: Configuring the NHRP Redirect and Shortcut Features on the Hub

```
Router(config)# interface tunnel 5
Router(config-if)# ipv6 address 2001:DB8:1:1::72/64

Router(config-if)# ipv6 nhrp redirect

Router(config-if)# ipv6 nhrp shortcut
```

Example: Configuring the Spoke for DMVPN

```
Router# configure terminal
Router (config)# crypto ikev2 keyring DMVPN
Router (config)# peer DMVPN
Router (config)# address 0.0.0.0 0.0.0.0
Router (config)# pre-shared-key cisco123
Router (config)# peer DMVPNv6
Router (config)# address ::/0
Router (config)# pre-shared-key cisco123v6
Router (config)# crypto ikev2 profile DMVPN
Router (config)# match identity remote address 0.0.0.0
Router (config)# match identity remote address ::/0
Router (config)# authentication local pre-share
Router (config)# authentication remote pre-share
Router (config)# keyring DMVPN
Router (config)# dpd 30 5 on-demand
Router (config)# crypto ipsec transform-set DMVPN esp-aes esp-sha-hmac
Router (config)# mode transport
Router (config)# crypto ipsec profile DMVPN
Router (config)# set transform-set DMVPN
Router (config)# set ikev2-profile DMVPN
Router(config)# interface tunnel 5

Router(config-if)# bandwidth 1000
Router(config-if)# ip address 10.0.0.11 255.255.255.0
Router(config-if)# ip mtu 1400
Router(config-if)# ip nhrp authentication test
Router(config-if)# ip nhrp network-id 100000
Router(config-if)# ip nhrp nhs 10.0.0.1 nbma 2001:DB8:0:FFFF:1::1 multicast
Router(config-if)# vip nhrp shortcut
Router(config-if)# delay 1000
Router(config-if)# ipv6 address 2001:DB8:0:100::B/64
Router(config-if)# ipv6 mtu 1400
Router(config-if)# ipv6 nd ra mtu suppress
Router(config-if)# no ipv6 redirects
Router(config-if)# ipv6 eigrp 1
Router(config-if)# ipv6 nhrp authentication testv6
Router(config-if)# ipv6 nhrp network-id 100006
Router(config-if)# ipv6 nhrp nhs 2001:DB8:0:100::1 nbma 2001:DB8:0:FFFF:1::1 multicast
Router(config-if)# ipv6 nhrp shortcut
Router(config-if)# tunnel source Ethernet0/0
Router(config-if)# tunnel mode gre multipoint ipv6
Router(config-if)# tunnel key 100000
Router(config-if)# end
.
.
```

The following sample output is from the **show dmvpn** command, with the **ipv6** and **detail** keywords, for the spoke:

```
Legend: Attrb --> S - Static, D - Dynamic, I - Incomplete
        N - NATed, L - Local, X - No Socket
        # Ent --> Number of NHRP entries with same NBMA peer
        NHS Status: E --> Expecting Replies, R --> Responding
        UpDn Time --> Up or Down Time for a Tunnel
==========================================================================

Interface Tunnel1 is up/up, Addr. is 10.0.0.1, VRF ""
   Tunnel Src./Dest. addr: 192.169.2.10/MGRE, Tunnel VRF ""
   Protocol/Transport: "multi-GRE/IP", Protect "test_profile"

IPv6 NHS: 2001::6 RE
Type:Spoke, Total NBMA Peers (v4/v6): 1
    1.Peer NBMA Address: 192.169.2.9
        Tunnel IPv6 Address: 2001::6
        IPv6 Target Network: 2001::/112
        # Ent: 2, Status: NHRP, UpDn Time: never, Cache Attrib: S

IPv6 NHS: 2001::6 RE
Type:Unknown, Total NBMA Peers (v4/v6): 1
    2.Peer NBMA Address: 192.169.2.9
        Tunnel IPv6 Address: FE80::1
        IPv6 Target Network: FE80::1/128
        # Ent: 0, Status: UP, UpDn Time: 00:00:24, Cache Attrib: D

Pending DMVPN Sessions:

Interface: Tunnel1
  IKE SA: local 192.169.2.10/500 remote 192.169.2.9/500 Active
  Crypto Session Status: UP-ACTIVE
  fvrf: (none), Phase1_id: 192.169.2.9
  IPSEC FLOW: permit 47 host 192.169.2.10 host 192.169.2.9
        Active SAs: 2, origin: crypto map
   Outbound SPI : 0x6F75C431, transform : esp-3des esp-sha-hmac
    Socket State: Open
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 supported feature list	"Start Here: Cisco IOS Software Release Specifics for IPv6 Features" module of the *Cisco IOS IPv6 Configuration Guide*
IPv6 IPsec	"Implementing IPsec in IPv6 Security" module of the *Cisco IOS IPv6 Configuration Guide*
IPv6 basic connectivity	"Implementing IPv6 Addressing and Basic Connectivity" module of the *Cisco IOS IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*

Related Topic	Document Title
DMVPN implementation for IPv4	"Dynamic Multipoint VPN (DMVPN)" module of the *Cisco IOS Security Configuration Guide*
DMVPN commands for IPv4	*Cisco IOS Security Command Reference*
NHRP for IPv4	"Configuring NHRP" module of the *Cisco IOS IP Addressing Services Configuration Guide*
NHRP commands for IPv4	The "NHRP Commands" section of the *Cisco IOS IP Addressing Services Command Reference*

Standards

Standard	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIB	MIBs Link
Cisco NHRP Extension MIB	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2332	*NBMA Next Hop Resolution Protocol (NHRP)*
RFC 2677	Managed Objects for the NBMA Next Hop Resolution Protocol (NHRP)

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing DMVPN for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 14 Feature Information for Implementing DMVPN for IPv6

Feature Name	Releases	Feature Information
DMVPN for IPv6	12.4(20)T	The Dynamic Multipoint VPN feature allows users to better scale large and small IPsec Virtual Private Networks by combining generic routing encapsulation tunnels, IPsec encryption, and NHRP. In DMVPN for IPv6, the public network (the Internet) is a pure IPv4 network, and the private network (the intranet) is IPv6 capable.
mGRE over IPV6	15.2(1)T	

Feature Name	Releases	Feature Information
IPv6 transport for DMVPN	15.2(1)T	The IPv6 transport for DMVPN feature builds IPv6 WAN-side capability into NHRP tunnels and the underlying IPsec encryption, and enables IPv6 to transport payloads on the Internet. The IPv6 transport for DMVPN feature is enabled by default.

Implementing EIGRP for IPv6

Customers can configure Enhanced Interior Gateway Routing Protocol (EIGRP) to route IPv6 prefixes. EIGRP IPv4 runs over an IPv4 transport, communicates only with IPv4 peers, and advertises only IPv4 routes, and EIGRP for IPv6 follows the same model. EIGRP for IPv4 and EIGRP for IPv6 are configured and managed separately. However, the configuration of EIGRP for IPv4 and IPv6 is similar and provides operational familiarity and continuity.

This document provides information about configuring and implementing EIGRP for IPv6.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Implementing EIGRP for IPv6

This section lists ways in which EIGRP for IPv6 differs from EIGRP IPv4 and lists EIGRP for IPv6 restrictions:

- EIGRP for IPv6 is directly configured on the interfaces over which it runs. This feature allows EIGRP for IPv6 to be configured without the use of a global IPv6 address. There is no network statement in EIGRP for IPv6.

In per-interface configuration at system startup, if EIGRP has been configured on an interface, then the EIGRP protocol may start running before any EIGRP router mode commands have been executed.

- An EIGRP for IPv6 protocol instance requires a router ID before it can start running.

- EIGRP for IPv6 has a shutdown feature. The routing process should be in "no shut" mode in order to start running.
- When a user uses a passive-interface configuration, EIGRP for IPv6 need not be configured on the interface that is made passive.
- EIGRP for IPv6 provides route filtering using the distribute-list prefix-list command. Use of the route-map command is not supported for route filtering with a distribute list.

Information About Implementing EIGRP for IPv6

Cisco EIGRP for IPv6 Implementation

EIGRP is an enhanced version of the IGRP developed by Cisco. EIGRP uses the same distance vector algorithm and distance information as IGRP. However, the convergence properties and the operating efficiency of EIGRP have improved substantially over IGRP.

The convergence technology is based on research conducted at SRI International and employs an algorithm called the diffusing update algorithm (DUAL). This algorithm guarantees loop-free operation at every instant throughout a route computation and allows all devices involved in a topology change to synchronize at the same time. Devices that are not affected by topology changes are not involved in recomputations. The convergence time with DUAL rivals that of any other existing routing protocol.

EIGRP provides the following features:

- Increased network width--With Routing Information Protocol (RIP), the largest possible width of your network is 15 hops. When EIGRP is enabled, the largest possible width is 224 hops. Because the EIGRP metric is large enough to support thousands of hops, the only barrier to expanding the network is the transport layer hop counter. Cisco works around this limitation by incrementing the transport control field only when an IPv4 or an IPv6 packet has traversed 15 devices and the next hop to the destination was learned by way of EIGRP. When a RIP route is being used as the next hop to the destination, the transport control field is incremented as usual.
- Fast convergence--The DUAL algorithm allows routing information to converge as quickly as any other routing protocol.
- Partial updates--EIGRP sends incremental updates when the state of a destination changes, instead of sending the entire contents of the routing table. This feature minimizes the bandwidth required for EIGRP packets.
- Neighbor discovery mechanism--This is a simple hello mechanism used to learn about neighboring devices. It is protocol-independent.
- Arbitrary route summarization.
- Scaling--EIGRP scales to large networks.
- Route filtering--EIGRP for IPv6 provides route filtering using the **distribute-list prefix-list**command. Use of the **route-map** command is not supported for route filtering with a distribute list.

EIGRP has the following four basic components:

- Neighbor discovery--Neighbor discovery is the process that devices use to dynamically learn of other devices on their directly attached networks. Devices must also discover when their neighbors become unreachable or inoperative. EIGRP neighbor discovery is achieved with low overhead by periodically sending small hello packets. EIGRP neighbors can also discover a neighbor that has recovered after an outage because the recovered neighbor will send out a hello packet. As long as hello packets are

received, the Cisco software can determine that a neighbor is alive and functioning. Once this status is determined, the neighboring devices can exchange routing information.

- Reliable transport protocol--The reliable transport protocol is responsible for guaranteed, ordered delivery of EIGRP packets to all neighbors. It supports intermixed transmission of multicast and unicast packets. Some EIGRP packets must be sent reliably and others need not be. For efficiency, reliability is provided only when necessary. For example, on a multiaccess network that has multicast capabilities, it is not necessary to send hello packets reliably to all neighbors individually. Therefore, EIGRP sends a single multicast hello with an indication in the packet informing the receivers that the packet need not be acknowledged. Other types of packets (such as updates) require acknowledgment, which is indicated in the packet. The reliable transport has a provision to send multicast packets quickly when unacknowledged packets are pending. This provision helps to ensure that convergence time remains low in the presence of varying speed links.
- DUAL finite state machine--The DUAL finite state machine embodies the decision process for all route computations. It tracks all routes advertised by all neighbors. DUAL uses several metrics including distance and cost information to select efficient, loop-free paths. When multiple routes to a neighbor exist, DUAL determines which route has the lowest metric (named the feasible distance), and enters this route into the routing table. Other possible routes to this neighbor with larger metrics are received, and DUAL determines the reported distance to this network. The reported distance is defined as the total metric advertised by an upstream neighbor for a path to a destination. DUAL compares the reported distance with the feasible distance, and if the reported distance is less than the feasible distance, DUAL considers the route to be a feasible successor and enters the route into the topology table. The feasible successor route that is reported with the lowest metric becomes the successor route to the current route if the current route fails. To avoid routing loops, DUAL ensures that the reported distance is always less than the feasible distance for a neighbor device to reach the destination network; otherwise, the route to the neighbor may loop back through the local device.
- Protocol-dependent modules--When there are no feasible successors to a route that has failed, but there are neighbors advertising the route, a recomputation must occur. This is the process in which DUAL determines a new successor. The amount of time required to recompute the route affects the convergence time. Recomputation is processor-intensive; it is advantageous to avoid unneeded recomputation. When a topology change occurs, DUAL will test for feasible successors. If there are feasible successors, DUAL will use them in order to avoid unnecessary recomputation.

The protocol-dependent modules are responsible for network layer protocol-specific tasks. For example, the EIGRP module is responsible for sending and receiving EIGRP packets that are encapsulated in IPv4 or IPv6. It is also responsible for parsing EIGRP packets and informing DUAL of the new information received. EIGRP asks DUAL to make routing decisions, but the results are stored in the IPv4 or IPv6 routing table. Also, EIGRP is responsible for redistributing routes learned by other IPv4 or IPv6 routing protocols.

How to Implement EIGRP for IPv6

Enabling EIGRP for IPv6 on an Interface

EIGRP for IPv6 is directly configured on the interfaces over which it runs, which allows EIGRP for IPv6 to be configured without the use of a global IPv6 address.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **interface** *type number*
5. **no shut**
6. **ipv6 enable**
7. **ipv6 eigrp** *as-number*
8. **ipv6 router eigrp** *as-number*
9. **router-id** *ip-address*
10. **exit**
11. **show ipv6 eigrp** [*as-number*] **interfaces** [*type number*] [**detail**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.

	Command or Action	Purpose
Step 4	**interface** *type number* **Example:** `Device(config)# interface FastEthernet 0/0`	Specifies the interface on which EIGRP is to be configured.
Step 5	**no shut** **Example:** `Device(config-if)# no shut`	Enables no shut mode so the routing process can start running.
Step 6	**ipv6 enable** **Example:** `Device(config-if)# ipv6 enable`	Enables IPv6 processing on an interface that has not been configured with an explicit IPv6 address.
Step 7	**ipv6 eigrp** *as-number* **Example:** `Device(config-if)# ipv6 eigrp 1`	Enables EIGRP for IPv6 on a specified interface.
Step 8	**ipv6 router eigrp** *as-number* **Example:** `Device(config-if)# ipv6 router eigrp 1`	Enters router configuration mode and creates an EIGRP IPv6 routing process.
Step 9	**router-id** *ip-address* **Example:** `Device(config-router)# router-id 10.1.1.1`	Enables the use of a fixed router ID. Use this command only if an IPv4 address is not defined on the router eligible for router ID.
Step 10	**exit** **Example:** `Device(config-router)# exit`	Enter three times to return to privileged EXEC mode.

	Command or Action	Purpose
Step 11	**show ipv6 eigrp** [*as-number*] **interfaces** [*type number*] [**detail**] **Example:** `Device# show ipv6 eigrp interfaces`	Displays information about interfaces configured for EIGRP for IPv6 .

Configuring the Percentage of Link Bandwidth Used by EIGRP

By default, EIGRP packets consume a maximum of 50 percent of the link bandwidth, as configured with the **bandwidth interface** command. You might want to change that value if a different level of link utilization is required or if the configured bandwidth does not match the actual link bandwidth (it may have been configured to influence route metric calculations).

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no shut**
5. **ipv6 bandwidth-percent eigrp** *as-number percent*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface FastEthernet 0/0`	Specifies the interface on which EIGRP is configured.

	Command or Action	Purpose
Step 4	**no shut** **Example:** `Device(config)# no shut`	Enables no shut mode so the routing process can start running.
Step 5	**ipv6 bandwidth-percent eigrp** *as-number percent* **Example:** `Device(config-if)# ipv6 bandwidth-percent eigrp 1 75`	Configures the percentage of bandwidth that may be used by EIGRP for IPv6 on an interface

Configuring Summary Addresses

This task configures a summary address for a specified interface. If other specific routes are in the routing table, EIGRP for IPv6 will advertise the summary address out the interface with a metric equal to the minimum of all more specific routes.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no shut**
5. **ipv6 summary-address eigrp** *as-number ipv6-address* [*admin-distance*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Device(config)# interface FastEthernet 0/0`	Specifies the interface on which EIGRP is configured.
Step 4	**no shut** **Example:** `Device(config)# no shut`	Enables no shut mode so the routing process can start running.
Step 5	**ipv6 summary-address eigrp** *as-number ipv6-address* [*admin-distance*] **Example:** `Device(config-if)# ipv6 summary-address eigrp 1 2001:DB8:0:1::/64`	Configures a summary aggregate address for a specified interface.

Configuring EIGRP Route Authentication

EIGRP route authentication provides message digest algorithm 5 (MD5) authentication of routing updates from the EIGRP routing protocol. The MD5 keyed digest in each EIGRP packet prevents the introduction of unauthorized or false routing messages from unapproved sources.

Each key has its own key identifier, which is stored locally. The combination of the key identifier and the interface associated with the message uniquely identifies the authentication algorithm and MD5 authentication key in use.

You can configure multiple keys with lifetimes. Only one authentication packet is sent, regardless of how many valid keys exist. The software examines the key numbers in order from lowest to highest, and uses the first valid key it encounters. Note that the device needs to know the time.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no shut**
5. **ipv6 authentication mode eigrp** *as-number* **md5**
6. **ipv6 authentication key-chain eigrp** *as-number key-chain*
7. **exit**
8. **key chain** *name-of-chain*
9. **key** *key-id*
10. **key-string** *text*
11. **accept-lifetime** *start-time* **infinite** | *end-time*| **duration** *seconds*
12. **send-lifetime** *start-time* **infinite** | *end-time* | **duration** *seconds*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface FastEthernet 0/0`	Specifies the interface on which EIGRP is configured.
Step 4	**no shut** **Example:** `Device(config)# no shutdown`	Enables no shut mode so the routing process can start running.

	Command or Action	Purpose
Step 5	**ipv6 authentication mode eigrp** *as-number* **md5** **Example:** `Device(config-if)# ipv6 authentication mode eigrp 1 md5`	Specifies the type of authentication used in EIGRP for IPv6 packets.
Step 6	**ipv6 authentication key-chain eigrp** *as-number key-chain* **Example:** `Device(config-if)# ipv6 authentication key-chain eigrp 1 chain1`	Enables authentication of EIGRP for IPv6 packets.
Step 7	**exit** **Example:** `Device(config-if)# exit`	Exits to global configuration mode.
Step 8	**key chain** *name-of-chain* **Example:** `Device(config)# key chain chain1`	Identifies a group of authentication keys. • Use the name specified in Step 5.
Step 9	**key** *key-id* **Example:** `Device(config-keychain)# key 1`	Identifies an authentication key on a key chain.
Step 10	**key-string** *text* **Example:** `Device(config-keychain-key)# key-string chain 1`	Specifies the authentication string for a key.
Step 11	**accept-lifetime** *start-time* **infinite** | *end-time*| **duration** *seconds* **Example:** `Device(config-keychain-key)# accept-lifetime 14:30:00 Jan 10 2006 duration 7200`	Sets the time period during which the authentication key on a key chain is received as valid.

	Command or Action	Purpose
Step 12	**send-lifetime** *start-time* **infinite** | *end-time* | **duration** *seconds* **Example:** `Device(config-keychain-key)# send-lifetime 15:00:00 Jan 10 2006 duration 3600`	Sets the time period during which an authentication key on a key chain is valid to be sent.

Overriding the Next Hop in EIGRP

EIGRP will, by default, set the IPv6 next-hop value to be itself for routes that it is advertising, even when advertising those routes back out the same interface where it learned them. Perform this task to change this default and instruct EIGRP to use the received next-hop value when advertising these routes.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no shut**
5. **no ipv6 next-hop-self eigrp** *as-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface FastEthernet 0/0`	Specifies the interface on which EIGRP is configured.

	Command or Action	Purpose
Step 4	**no shut** **Example:** `Device(config)# no shutdown`	Enables no shut mode so the routing process can start running.
Step 5	**no ipv6 next-hop-self eigrp** *as-number* **Example:** `Device(config-if)# no ipv6 next-hop-self eigrp 1`	Changes the default IPv6 next-hop value and instructs EIGRP to use the received next-hop value.

Adjusting the Interval Between Hello Packets in EIGRP for IPv6

Routing devices periodically send hello packets to each other to dynamically learn of other devices on their directly attached networks. This information is used to discover neighbors and learn when neighbors become unreachable or inoperative.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no shut**
5. **ipv6 hello-interval eigrp** *as-number seconds*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Device(config)# interface FastEthernet 0/0`	Specifies the interface on which EIGRP is configured.
Step 4	**no shut** **Example:** `Device(config)# no shutdown`	Enables no shut mode so the routing process can start running.
Step 5	**ipv6 hello-interval eigrp** *as-number seconds* **Example:** `Device(config)# ipv6 hello-interval eigrp 1 10`	Configures the hello interval for the EIGRP for IPv6 routing process designated by an autonomous system number.

Adjusting the Hold Time in EIGRP for IPv6

On very congested and large networks, the default hold time might not be sufficient time for all devices to receive hello packets from their neighbors. In this case, you may want to increase the hold time.

This task configures the hold time on a specified interface for a particular EIGRP routing process designated by the autonomous system number. The hold time is advertised in hello packets and indicates to neighbors the length of time they should consider the sender valid. The default hold time is 3 times the hello interval, or 15 seconds. For slow-speed nonbroadcast multi-access (NBMA) networks, the default hold time is 180 seconds. The hold time should be changed if the hello-interval value is changed.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no shut**
5. **ipv6 hold-time eigrp** *as-number seconds*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface FastEthernet 0/0`	Specifies the interface on which EIGRP is configured.
Step 4	**no shut** **Example:** `Device(config)# no shutdown`	Enables no shut mode so the routing process can start running.
Step 5	**ipv6 hold-time eigrp** *as-number seconds* **Example:** `Device(config)# ipv6 hold-time eigrp 1 40`	Configures the hold time for a particular EIGRP for IPv6 routing process designated by the autonomous system number.

Disabling Split Horizon in EIGRP for IPv6

By default, split horizon is enabled on all interfaces. Split horizon controls the sending of EIGRP update and query packets. When split horizon is enabled on an interface, update and query packets are not sent for destinations for which this interface is the next hop. Controlling update and query packets in this manner reduces the possibility of routing loops.

Split horizon blocks route information from being advertised by a device out of any interface from which that information originated. This behavior usually optimizes communications among multiple routing devices, particularly when links are broken. However, with nonbroadcast networks (such as multipoint GRE), situations can arise for which this behavior is not ideal. For these situations, including networks in which you have EIGRP configured, you may want to disable split horizon.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no shut**
5. **no ipv6 split-horizon eigrp** *as-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface FastEthernet 0/0`	Specifies the interface on which EIGRP is configured.
Step 4	**no shut** **Example:** `Device(config)# no shutdown`	Enables no shut mode so the routing process can start running.
Step 5	**no ipv6 split-horizon eigrp** *as-number* **Example:** `Device(config-if)# no ipv6 split-horizon eigrp 101`	Disables EIGRP for IPv6 split horizon on the specified interface.

Configuring EIGRP Stub Routing for Greater Network Stability

The EIGRP stub routing feature can help to provide greater network stability. In the event of network instability, this feature prevents EIGRP queries from being sent over limited bandwidth links to nontransit devices. Instead, distribution devices to which the stub device is connected answer the query on behalf of the stub device. This feature greatly reduces the chance of further network instability due to congested or problematic WAN links. The EIGRP stub routing feature also simplifies the configuration and maintenance of hub-and-spoke networks. When stub routing is enabled in dual-homed remote configurations, it is no longer necessary to configure filtering on remote devices to prevent those remote devices from appearing as transit paths to the hub devices.

Caution EIGRP stub routing should be used only on stub devices. A stub device is defined as a device connected to the network core or distribution layer through which core transit traffic should not flow. A stub device should not have any EIGRP neighbors other than distribution devices.

Configuring a Device for EIGRP Stub Routing

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router eigrp** *as-number*
4. **eigrp stub receive-only** | **leak-map** | **connected** | **static** | **summary** | **redistributed**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router eigrp** *as-number* **Example:** `Device(config)# ipv6 router eigrp 1`	Specifies the EIGRP for IPv6 routing process to be configured.
Step 4	**eigrp stub receive-only** | **leak-map** | **connected** | **static** | **summary** | **redistributed** **Example:** `Device(config-router)# eigrp stub`	Configures a device as a stub using EIGRP.

Verifying EIGRP Stub Routing

SUMMARY STEPS

1. **enable**
2. **show ipv6 eigrp neighbors detail** *interface-type* | *as-number* | **static**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ipv6 eigrp neighbors detail** *interface-type* | *as-number* | **static** **Example:** `Device# show ipv6 eigrp neighbors detail`	Displays the neighbors discovered by EIGRP for IPv6. This command is performed on the distribution layer device to view the status of the remote device.

Customizing an EIGRP for IPv6 Routing Process

Logging EIGRP Neighbor Adjacency Changes

You can enable the logging of neighbor adjacency changes to monitor the stability of the routing system and to help you detect problems. By default, adjacency changes are logged.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router eigrp** *as-number*
4. **eigrp log-neighbor-changes**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router eigrp** *as-number* **Example:** `Device(config)# ipv6 router eigrp 1`	Specifies the EIGRP for IPv6 routing process to be configured.
Step 4	**eigrp log-neighbor-changes** **Example:** `Device(config-router)# eigrp log-neighbor-changes`	Enables the logging of changes in EIGRP for IPv6 neighbor adjacencies.

Configuring Intervals Between Neighbor Warnings

When neighbor warning messages occur, they are logged by default.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router eigrp** *as-number*
4. **eigrp log-neighbor-warnings** [*seconds*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router eigrp** *as-number* **Example:** `Device(config)# ipv6 router eigrp 1`	Specifies the EIGRP for IPv6 routing process to be configured.
Step 4	**eigrp log-neighbor-warnings** [*seconds*] **Example:** `Device(config-router)# eigrp log-neighbor-warnings 300`	Configures the logging intervals of EIGRP neighbor warning messages.

Adjusting EIGRP for IPv6 Metric Weights

EIGRP for IPv6 uses the minimum bandwidth on the path to a destination network and the total delay to compute routing metrics. You can use the **metric weights** command to adjust the default behavior of EIGRP for IPv6 routing and metric computations. EIGRP for IPv6 metric defaults have been carefully selected to provide optimal performance in most networks.

Note Adjusting EIGRP metric weights can dramatically affect network performance. Because of the complexity of this task, we recommend that you do not change the default values without guidance from an experienced network designer.

By default, the EIGRP composite metric is a 32-bit quantity that is a sum of the segment delays and the lowest segment bandwidth (scaled and inverted) for a given route. For a network of homogeneous media, this metric reduces to a hop count. For a network of mixed media (e.g., GigabitEthernet, FastEthernet, Ethernet), the route with the lowest metric reflects the most desirable path to a destination.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router eigrp** *as-number*
4. **metric weights** *tos k1 k2 k3 k4 k5*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router eigrp** *as-number* **Example:** `Device(config)# ipv6 router eigrp 1`	Specifies the EIGRP for IPv6 routing process to be configured.
Step 4	**metric weights** *tos k1 k2 k3 k4 k5* **Example:** `Device(config-router)# metric weights 0 2 0 2 0 0`	Tunes EIGRP metric calculations.

Deleting Entries from EIGRP for IPv6 Routing Tables

SUMMARY STEPS

1. **enable**
2. **clear ipv6 eigrp** [*as-number*] [**neighbor** [*ipv6-address* | *interface-type interface-number*]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

Command or Action	Purpose
Step 2 **clear ipv6 eigrp** [*as-number*] [**neighbor** [*ipv6-address* | *interface-type interface-number*]] **Example:** `Device# clear ipv6 eigrp neighbor 3FEE: 12E1:2AC1:EA32`	Deletes entries from EIGRP for IPv6 routing tables. The routes that are cleared are the routes that were learned by the specified device.

Configuration Examples for Implementing EIGRP for IPv6

- Example: Configuring EIGRP to Establish Adjacencies on an Interface, page 283

Example: Configuring EIGRP to Establish Adjacencies on an Interface

EIGRP for IPv6 is configured directly on the interfaces over which it runs. This example shows the minimal configuration required for EIGRP for IPv6 to send hello packets in order to establish adjacencies on Ethernet 0/0:

```
ipv6 unicast-routing
interface ethernet0/0
no shut
  ipv6 enable
  ipv6 eigrp 1
!
ipv6 router eigrp 1
  eigrp router-id 10.1.1.1
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 supported feature list	Start Here: Cisco IOS Software Release Specifics for IPv6 Features
Implementing IS-IS for IPv6	Implementing IS-IS for IPv6
Implementing Multiprotocol BGP for IPv6	Implementing Multiprotocol BGP for IPv6
Implementing RIP for IPv6	Implementing RIP for IPv6
EIGRP for IPv4	"Configuring EIGRP," *Cisco IOS IP Routing Protocols Configuration Guide*
EIGRP for IPv6 commands	*Cisco IOS IPv6 Command Reference*

Related Topic	Document Title
EIGRP for IPv4 commands	"EIGRP Commands," *Cisco IOS IP Routing Protocols Command Reference*

Standards

Standard	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing EIGRP for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software

release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 15 Feature Information for Implementing EIGRP for IPv6

Feature Name	Releases	Feature Information
IPv6 Routing: EIGRP Support	12.4(6)T 12.2(33)SRB 12.2(33)SXI 15.0(1)S	Customers can configure EIGRP to route IPv6 prefixes. There is no linkage between EIGRP for IPv4 and EIGRP for IPv6; they are configured and managed separately. However, the configuration of EIGRP for IPv4 and IPv6 is similar and provides operational familiarity and continuity. The following commands were introduced or modified for this feature: **accept-lifetime**, clear ipv6 eigrp, **eirgp log-neighbor-changes**, **eigrp log-neighbor-warnings**, **eigrp router-id**, **eigrp stub**, ipv6 authentication key-chain eigrp, **ipv6 authentication mode eigrp**, **ipv6 bandwidth-percent eigrp**, **ipv6 eigrp**, **ipv6 hello-interval eigrp**, **ipv6 hold-time eigrp**, **ipv6 next-hop-self eigrp**, **ipv6 router eigrp**, **ipv6 split-horizon eigrp**, **ipv6 summary-address eigrp**, **ipv6 unicast-routing**, **key**, **key chain**, **key-string**, **metric weights**, **send-lifetime**, **show ipv6 eigrp**, show ipv6 eigrp neighbors

Feature Name	Releases	Feature Information
EIGRP IPv6 VRF Lite	15.0(1)SY1 15.1(1)S	The EIGRP IPv6 VRF Lite feature provides EIGRP IPv6 support for multiple VRFs. EIGRP for IPv6 can operate in the context of a VRF. The EIGRP IPv6 VRF Lite feature provides separation between routing and forwarding, providing an additional level of security because no communication between devices belonging to different VRFs is allowed unless it is explicitly configured. The EIGRP IPv6 VRF Lite feature simplifies the management and troubleshooting of traffic belonging to a specific VRF. The EIGRP IPv6 VRF Lite feature is available only in EIGRP named configurations. There are no new or modified commands for this feature.

Configuring First Hop Redundancy Protocols in IPv6

IPv6 routing protocols ensure router-to-router resilience and failover. However, in situations in which the path between a host and the first-hop router fails, or the first-hop router itself fails, first hop redundancy protocols (FHRPs) ensure host-to-router resilience and failover.

The Gateway Load Balancing Protocol (GLBP) FHRP protects data traffic from a failed router or circuit, while allowing packet load sharing between a group of redundant routers. The Hot Standby Router Protocol (HSRP) protects data traffic in case of a gateway failure.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for First Hop Redundancy Protocols in IPv6

- Before configuring GLBP, ensure that the routers can support multiple MAC addresses on the physical interfaces. An additional MAC address is used for each GLBP forwarder to be configured.
- Avoid static link-local addressing on interfaces configured with GLBP.
- HSRP version 2 must be enabled on an interface before HSRP IPv6 can be configured.

Information About First Hop Redundancy Protocols in IPv6

GLBP for IPv6

GLBP for IPv6 Overview

The Gateway Load Balancing Protocol feature provides automatic router backup for IPv6 hosts configured with a single default gateway on an IEEE 802.3 LAN. Multiple first hop routers on the LAN combine to offer a single virtual first-hop IPv6 router while sharing the IPv6 packet forwarding load. GLBP performs a similar function for the user as HSRP. HSRP allows multiple routers to participate in a virtual router group configured with a virtual IPv6 address. One member is elected to be the active router to forward packets sent to the virtual IPv6 address for the group. The other routers in the group are redundant until the active router fails. These standby routers have unused bandwidth that the protocol is not using. Although multiple virtual router groups can be configured for the same set of routers, the hosts must be configured for different default gateways, which results in an extra administrative burden. The advantage of GLBP is that it additionally provides load balancing over multiple routers (gateways) using a single virtual IPv6 address and multiple virtual MAC addresses. The forwarding load is shared among all routers in a GLBP group rather than being handled by a single router while the other routers stand idle. Each host is configured with the same virtual IPv6 address, and all routers in the virtual router group participate in forwarding packets.

GLBP Benefits

GLBP for IPv6 provides the following benefits:

Load Sharing

You can configure GLBP in such a way that traffic from LAN clients can be shared equitably among multiple routers.

Multiple Virtual Routers

GLBP supports up to 1024 virtual routers (GLBP groups) on each physical interface of a router and up to four virtual forwarders per group.

Preemption

The redundancy scheme of GLBP enables you to preempt an active virtual gateway with a higher priority backup virtual gateway that has become available. Forwarder preemption works in a similar way, except that forwarder preemption uses weighting instead of priority and is enabled by default.

Authentication

You can also use the industry-standard Message Digest algorithm 5 (MD5) algorithm for improved reliability, security, and protection against GLBP-spoofing software. A router within a GLBP group with a different authentication string than other routers will be ignored by other group members. You can alternatively use a simple text password authentication scheme between GLBP group members to detect configuration errors.

GLBP Active Virtual Gateway

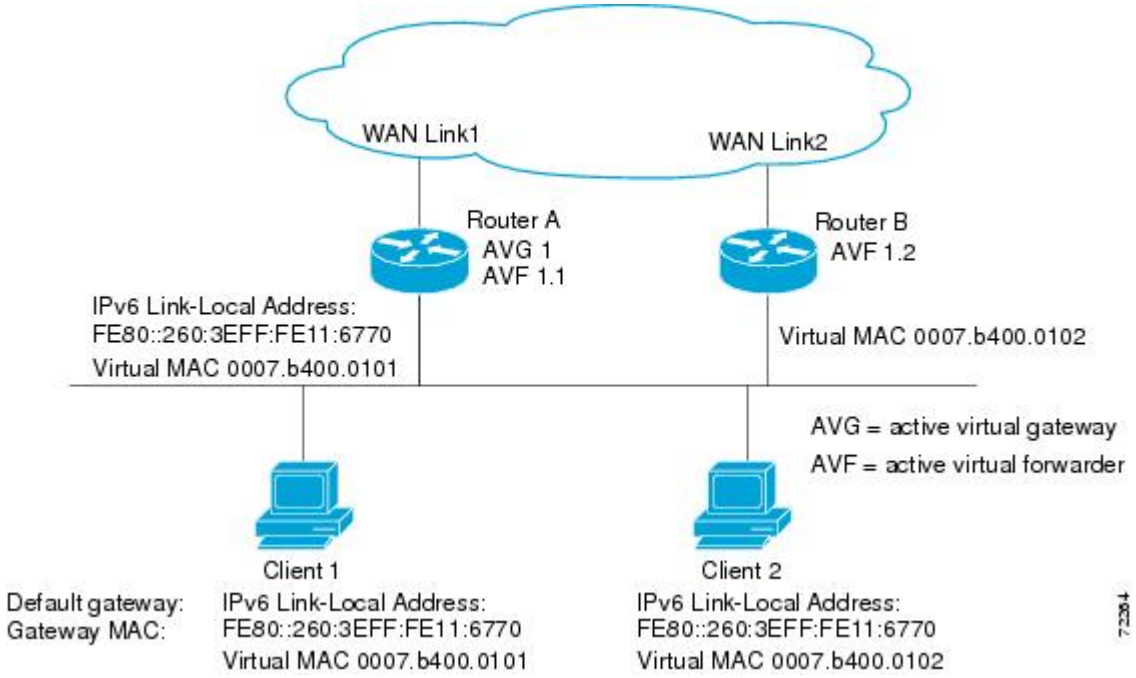
Members of a GLBP group elect one gateway to be the active virtual gateway (AVG) for that group. Other group members provide backup for the AVG in the event that the AVG becomes unavailable. The function of the AVG is that it assigns a virtual MAC address to each member of the GLBP group. Each gateway assumes responsibility for forwarding packets sent to the virtual MAC address assigned to it by the AVG. These gateways are known as active virtual forwarders (AVFs) for their virtual MAC address.

The IPv6 neighbor discovery process uses Internet Control Message Protocol (ICMP) in IPv6 messages and solicited-node multicast addresses to determine the link-layer address of a neighbor on the same network (local link), verify the reachability of a neighbor, and track neighboring routers. The IPv6 neighbor discovery process uses ICMPv6 messages and solicited-node multicast addresses to determine the link-layer address of a neighbor on the same network (local link), verify the reachability of a neighbor, and track neighboring routers. The AVG is responsible for answering ICMPv6 Neighbor Discovery requests for the virtual IPv6 address. Load sharing is achieved by the AVG replying to the ICMPv6 Neighbor Discovery requests with different virtual MAC addresses.

In the figure below, Router A is the AVG for a GLBP group, and is responsible for the IPv6 link-local address FE80::260:3EFF:FE11:6770. Router A is also an AVF for the virtual MAC address 0007.b400.0101. Router B is a member of the same GLBP group and is designated as the AVF for the virtual MAC address 0007.b400.0102. Client 1 has a default gateway IPv6 address of FE80::260:3EFF:FE11:6770 and a gateway MAC address of 0007.b400.0101. Client 2 shares the same

default gateway IP address but receives the gateway MAC address 0007.b400.0102 because Router B is sharing the traffic load with Router A.

Figure 23 GLBP Topology



If Router A becomes unavailable, Client 1 will not lose access to the WAN because Router B will assume responsibility for forwarding packets sent to the virtual MAC address of Router A, and for responding to packets sent to its own virtual MAC address. Router B will also assume the role of the AVG for the entire GLBP group. Communication for the GLBP members continues despite the failure of a router in the GLBP group.

GLBP Virtual MAC Address Assignment

A GLBP group allows up to four virtual MAC addresses per group. The AVG is responsible for assigning the virtual MAC addresses to each member of the group. Other group members request a virtual MAC address after they discover the AVG through hello messages. Gateways are assigned the next MAC address in sequence. A virtual forwarder that is assigned a virtual MAC address by the AVG is known as a primary virtual forwarder. Other members of the GLBP group learn the virtual MAC addresses from hello messages. A virtual forwarder that has learned the virtual MAC address is referred to as a secondary virtual forwarder.

GLBP Virtual Gateway Redundancy

GLBP operates virtual gateway redundancy in the same way as HSRP. One gateway is elected as the AVG, another gateway is elected as the standby virtual gateway, and the remaining gateways are placed in a listen state.

If an AVG fails, the standby virtual gateway will assume responsibility for the virtual IPv6 address. A new standby virtual gateway is then elected from the gateways in the listen state.

GLBP Virtual Forwarder Redundancy

GLBP virtual forwarder redundancy is similar to virtual gateway redundancy with an AVF. If the AVF fails, one of the secondary virtual forwarders in the listen state assumes responsibility for the virtual MAC address.

The new AVF is also a primary virtual forwarder for a different forwarder number. GLBP migrates hosts away from the old forwarder number using two timers that start as soon as the gateway changes to the active virtual forwarder state. GLBP uses the hello messages to communicate the current state of the timers.

The redirect time is the interval during which the AVG continues to redirect hosts to the old virtual forwarder MAC address. When the redirect time expires, the AVG stops using the old virtual forwarder MAC address in ICMPv6 ND replies, although the virtual forwarder will continue to forward packets that were sent to the old virtual forwarder MAC address.

The secondary hold time is the interval during which the virtual forwarder is valid. When the secondary hold time expires, the virtual forwarder is removed from all gateways in the GLBP group. The expired virtual forwarder number becomes eligible for reassignment by the AVG.

GLBP Gateway Priority

GLBP gateway priority determines the role that each GLBP gateway plays and the results if the AVG fails.

Priority also determines if a GLBP router functions as a backup virtual gateway and the order of ascendancy to becoming an AVG if the current AVG fails. You can configure the priority of each backup virtual gateway with a value of 1 through 255 using the **glbp priority** command.

In the figure above, if Router A--the AVG in a LAN topology--fails, an election process takes place to determine which backup virtual gateway should take over. In this example, Router B is the only other member in the group so it will automatically become the new AVG. If another router existed in the same GLBP group with a higher priority, then the router with the higher priority would be elected. If both routers have the same priority, the backup virtual gateway with the higher IPv6 address would be elected to become the active virtual gateway.

By default, the GLBP virtual gateway preemptive scheme is disabled. A backup virtual gateway can become the AVG only if the current AVG fails, regardless of the priorities assigned to the virtual gateways. You can enable the GLBP virtual gateway preemptive scheme using the **glbp preempt** command. Preemption allows a backup virtual gateway to become the AVG, if the backup virtual gateway is assigned a higher priority than the current AVG.

GLBP Gateway Weighting and Tracking

GLBP uses a weighting scheme to determine the forwarding capacity of each router in the GLBP group. The weighting assigned to a router in the GLBP group can be used to determine whether it will forward packets and, if so, the proportion of hosts in the LAN for which it will forward packets. Thresholds can be set to disable forwarding when the weighting falls below a certain value. When the weighting rises above another threshold, forwarding is automatically reenabled.

The GLBP group weighting can be automatically adjusted by tracking the state of an interface within the router. If a tracked interface goes down, the GLBP group weighting is reduced by a specified value. Different interfaces can be tracked to decrement the GLBP weighting by varying amounts.

By default, the GLBP virtual forwarder preemptive scheme is enabled with a delay of 30 seconds. A backup virtual forwarder can become the AVF if the current AVF weighting falls below the low weighting threshold for 30 seconds. You can disable the GLBP forwarder preemptive scheme using the **no glbp**

forwarder preempt command or change the delay using the **glbp forwarder preempt delay minimum** command.

HSRP for IPv6

IPv6 routing protocols ensure router-to-router resilience and failover. However, in situations in which the path between a host and the first-hop router fails, or the first-hop router itself fails, first hop redundancy protocols (FHRPs) ensure host-to-router resilience and failover.

The Hot Standby Router Protocol (HSRP) protects data traffic in case of a gateway failure.

HSRP for IPv6 Overview

The HSRP is an FHRP designed to allow for transparent failover of the first-hop IP router. HSRP provides high network availability by providing first-hop routing redundancy for IP hosts on Ethernet configured with a default gateway IP address. HSRP is used in a group of routers for selecting an active router and a standby router. In a group of router interfaces, the active router is the router of choice for routing packets; the standby router is the router that takes over when the active router fails or when preset conditions are met.

IPv6 hosts learn of available IPv6 routers through IPv6 neighbor discovery RA messages. These are multicast periodically, or may be solicited by hosts. HSRP is designed to provide only a virtual first hop for IPv6 hosts.

An HSRP IPv6 group has a virtual MAC address that is derived from the HSRP group number, and a virtual IPv6 link-local address that is, by default, derived from the HSRP virtual MAC address. Periodic RAs are sent for the HSRP virtual IPv6 link-local address when the HSRP group is active. These RAs stop after a final RA is sent when the group leaves the active state.

Periodic RAs for the interface link-local address stop after a final RA is sent while at least one virtual IPv6 link-local address is configured on the interface. No restrictions occur for the interface IPv6 link-local address other than that mentioned for the RAs. Other protocols continue to receive and send packets to this address.

HSRP uses a priority mechanism to determine which HSRP configured router is to be the default active router. To configure a router as the active router, you assign it a priority that is higher than the priority of all the other HSRP-configured routers. The default priority is 100, so if you configure just one router to have a higher priority, that router will be the default active router.

HSRP IPv6 Virtual MAC Address Range

HSRP IPv6 uses a different virtual MAC address block than does HSRP for IP:

0005.73A0.0000 through 0005.73A0.0FFF (4096 addresses)

HSRP IPv6 UDP Port Number

Port number 2029 has been assigned to HSRP IPv6.

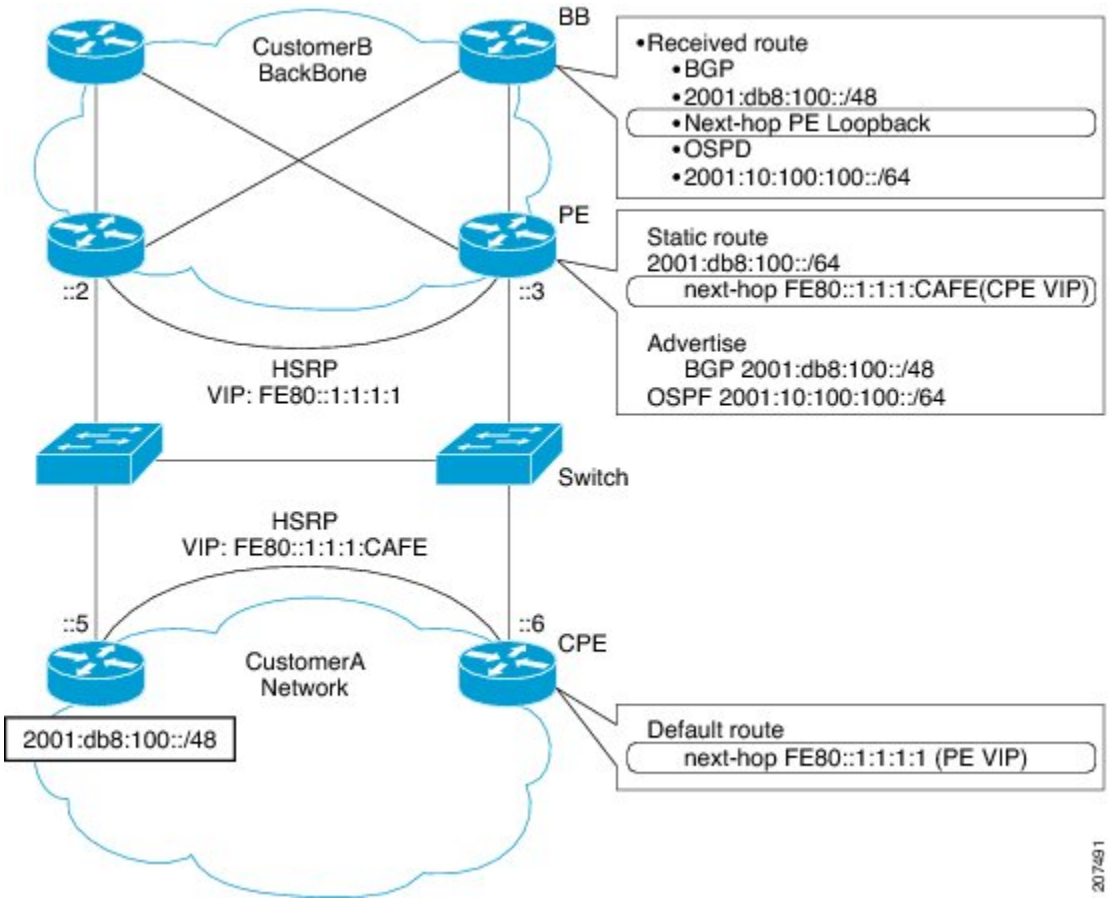
HSRP Global IPv6 Address

Note This feature is supported only in Cisco IOS Release 12.2(33)SXI4.

The HSRP global IPv6 address feature allows users to configure multiple nonlink local addresses as virtual addresses, and it allows for the storage and management of multiple global IPv6 virtual addresses in addition to the existing primary link-local address. If an IPv6 address is used, it must include an IPv6 prefix length. If a link-local address is used, it must not have a prefix.

The figure below depicts a deployment scenario that uses an HSRP IPv6 global virtual interface:

Figure 24 Scenario Using Gan HSRP IPv6 Global Virtual Interface



In the figure above, the provider equipment (PE) routers need to inject a route to reach the customer premises equipment (CPE) from the backbone routers. Because there are two CPEs, HSRP is convenient to use. The static route will be set with a link-local next hop (FE80::1:1:1:CAFE). If this address is injected in the backbone, this route is useless with a link-local next hop, as link-local addresses only have scope within the Layer 2 local LAN space. To address this issue, the next hop of the static route toward the virtual address must be set to a nonlink-local address, so backbone routers can route packets to the PE routers. At the next-hop address resolution, the active HSRP group member will reply to neighbor solicitation (NS) messages sent to the nonlink-local address.

How to Configure First Hop Redundancy Protocols in IPv6

Configuring and Customizing GLBP

Customizing GLBP behavior is optional. Be aware that as soon as you enable a GLBP group, that group is operating. It is possible that if you first enable a GLBP group before customizing GLBP, the router could take over control of the group and become the AVG before you have finished customizing the feature. Therefore, if you plan to customize GLBP, it is a good idea to do so before enabling GLBP.

Customizing GLBP

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 address** *ipv6-address / prefix-length | prefix-name ipv6-prefix / prefix-length* | **autoconfig** [*default-route*]}
5. **glbp** *group* **timers** [**msec**] *hellotime*[**msec**] *holdtime*
6. **glbp** *group* **timers redirect** *redirect timeout*
7. **glbp** *group* **load-balancing** [**host-dependent**| **round-robin**| **weighted**]
8. **glbp** *group* **priority** *level*
9. **glbp** *group* **preempt** [**delay minimum** *seconds*]
10. **glbp** *group* **name** *redundancy-name*
11. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface fastethernet 0/0`	Specifies an interface type and number, and enters interface configuration mode.
Step 4	**ipv6 address** *ipv6-address* / *prefix-length* | *prefix-name* *ipv6-prefix* / *prefix-length* | **autoconfig** [*default-route*]} **Example:** `Device(config-if)# ipv6 address 2001:DB8:0:7272::72/64`	Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.
Step 5	**glbp** *group* **timers** [**msec**] *hellotime*[**msec**] *holdtime* **Example:** `Device(config-if)# glbp 10 timers 5 18`	Configures the interval between successive hello packets sent by the AVG in a GLBP group.
Step 6	**glbp** *group* **timers redirect** *redirect timeout* **Example:** `Device(config-if)# glbp 10 timers redirect 600 7200`	Configures the time interval during which the AVG continues to redirect clients to an AVF. • The *timeout* argument specifies the interval in seconds before a secondary virtual forwarder becomes invalid.
Step 7	**glbp** *group* **load-balancing** [**host-dependent**| **round-robin**| **weighted**] **Example:** `Device(config-if)# glbp 10 load-balancing host-dependent`	Specifies the method of load balancing used by the GLBP AVG.
Step 8	**glbp** *group* **priority** *level* **Example:** `Device(config-if)# glbp 10 priority 254`	Sets the priority level of the gateway within a GLBP group. • The default value is 100.

	Command or Action	Purpose
Step 9	**glbp** *group* **preempt** [**delay minimum** *seconds*] **Example:** `Device(config-if)# glbp 10 preempt delay minimum 60`	Configures the router to take over as AVG for a GLBP group if it has a higher priority than the current AVG. • This command is disabled by default. • Use the optional **delay** and **minimum** keywords and the *seconds* argument to specify a minimum delay interval in seconds before preemption of the AVG takes place.
Step 10	**glbp** *group* **name** *redundancy-name* **Example:** `Device(config-if)# glbp 10 name abcompany`	Enables IPv6 redundancy by assigning a name to the GLBP group. • The GLBP redundancy client must be configured with the same GLBP group name so the redundancy client and the GLBP group can be connected.
Step 11	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode, and returns the router to global configuration mode.

Configuring GLBP Authentication

The following sections describe configuration tasks for GLBP authentication. The task you perform depends on whether you want to use text authentication, a simple MD5 key string, or MD5 key chains for authentication.

GLBP MD5 authentication provides greater security than the alternative plain text authentication scheme. MD5 authentication allows each GLBP group member to use a secret key to generate a keyed MD5 hash that is part of the outgoing packet. A keyed hash of an incoming packet is generated and, if the hash within the incoming packet does not match the generated hash, the packet is ignored.

The key for the MD5 hash can either be given directly in the configuration using a key string or supplied indirectly through a key chain.

A router will ignore incoming GLBP packets from routers that do not have the same authentication configuration for a GLBP group. GLBP has three authentication schemes:

- No authentication
- Plain text authentication
- MD5 authentication

GLBP packets will be rejected in any of the following cases:

- The authentication schemes differ on the router and in the incoming packet.
- MD5 digests differ on the router and in the incoming packet.
- Text authentication strings differ on the router and in the incoming packet.

- Configuring GLBP MD5 Authentication Using a Key String, page 297
- Configuring GLBP MD5 Authentication Using a Key Chain, page 298
- Configuring GLBP Text Authentication, page 301

Configuring GLBP MD5 Authentication Using a Key String

Configuring GLBP MD5 authentication protects the router against spoofing software and uses the industry-standard MD5 algorithm for improved reliability and security.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 address** *ipv6-address / prefix-length* | *prefix-name ipv6-prefix / prefix-length* | **autoconfig** [*default-route*]}
5. **glbp** *group-number* **authentication md5 key-string** [**0** | **7**] *key*
6. **glbp** *group* **ipv6** [*ipv6-address* | **autoconfig**
7. Repeat Steps 1 through 6 on each router that will communicate.
8. **end**
9. **show glbp**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface Ethernet 0/1`	Configures an interface type and enters interface configuration mode.
Step 4	**ipv6 address** *ipv6-address / prefix-length* | *prefix-name ipv6-prefix / prefix-length* | **autoconfig** [*default-route*]} **Example:** `Router(config-if)# ipv6 address 2001:DB8:0:7272::72/64`	Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.

Command or Action	Purpose
Step 5 **glbp** *group-number* **authentication md5 key-string** [**0** | **7**] *key* **Example:** `Router(config-if)# glbp 1 authentication md5 key-string d00b4r987654321a`	Configures an authentication key for GLBP MD5 authentication. • The number of characters in the command plus the key string must not exceed 255 characters. • No keyword before the *key* argument or specifying **0** means the key is unencrypted. • Specifying **7** means the key is encrypted. The key-string authentication key will automatically be encrypted if the **service password-encryption** global configuration command is enabled.
Step 6 **glbp** *group* **ipv6** [*ipv6-address* | **autoconfig** **Example:** `Router(config-if)# glbp 1 ipv6 FE80::260:3EFF:FE11:6770`	Enables GLBP in IPv6.
Step 7 Repeat Steps 1 through 6 on each router that will communicate.	--
Step 8 **end** **Example:** `Router(config-if)# end`	Returns to privileged EXEC mode.
Step 9 **show glbp** **Example:** `Router# show glbp`	(Optional) Displays GLBP information. • Use this command to verify your configuration. The key string and authentication type will be displayed if configured.

Configuring GLBP MD5 Authentication Using a Key Chain

Key chains allow a different key string to be used at different times according to the key chain configuration. GLBP will query the appropriate key chain to obtain the current live key and key ID for the specified key chain.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **key chain** *name-of-chain*
4. **key** *key-id*
5. **key-string** *string*
6. **exit**
7. **exit**
8. **interface** *type number*
9. **ipv6 address** *ipv6-address* / *prefix-length* | *prefix-name ipv6-prefix* / *prefix-length* | **autoconfig** [*default-route*]}
10. **glbp** *group-number* **authentication md5 key-chain** *name-of-chain*
11. **glbp** *group* **ipv6** [*ipv6-address* | **autoconfig**
12. Repeat Steps 1 through 11 on each router that will communicate.
13. **end**
14. **show glbp**
15. **show key chain**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**key chain** *name-of-chain* **Example:** `Router(config)# key chain glbp2`	Enables authentication for routing protocols and identifies a group of authentication keys.
Step 4	**key** *key-id* **Example:** `Router(config-keychain)# key 100`	Identifies an authentication key on a key chain. • The *key-id* must be a number.

	Command or Action	Purpose
Step 5	**key-string** *string* **Example:** `Router(config-keychain-key)# key-string string1`	Specifies the authentication string for a key. • The *string* can be 1 to 80 uppercase or lowercase alphanumeric characters; the first character cannot be a numeral.
Step 6	**exit** **Example:** `Router(config-keychain-key)# exit`	Returns to keychain configuration mode.
Step 7	**exit** **Example:** `Router(config-keychain)# exit`	Returns to global configuration mode.
Step 8	**interface** *type number* **Example:** `Router(config)# interface Ethernet 0/1`	Configures an interface type and enters interface configuration mode.
Step 9	**ipv6 address** *ipv6-address* / *prefix-length* | *prefix-name ipv6-prefix* / *prefix-length* | **autoconfig** [*default-route*]} **Example:** `Router(config-if)# ipv6 address 2001:DB8:0:7272::72/64`	Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.
Step 10	**glbp** *group-number* **authentication md5 key-chain** *name-of-chain* **Example:** `Router(config-if)# glbp 1 authentication md5 key-chain glbp2`	Configures an authentication MD5 key chain for GLBP MD5 authentication. • The key chain name must match the name specified in Step 3.
Step 11	**glbp** *group* **ipv6** [*ipv6-address* | **autoconfig** **Example:** `Router(config-if)# glbp 1 ipv6 FE80::E0:F727:E400:A`	Enables GLBP in IPv6.

	Command or Action	Purpose
Step 12	Repeat Steps 1 through 11 on each router that will communicate.	--
Step 13	**end** **Example:** `Router(config-if)# end`	Returns to privileged EXEC mode.
Step 14	**show glbp** **Example:** `Router# show glbp`	(Optional) Displays GLBP information. • Use this command to verify your configuration. The key chain and authentication type will be displayed if configured.
Step 15	**show key chain** **Example:** `Router# show key chain`	(Optional) Displays authentication key information.

Configuring GLBP Text Authentication

This method of authentication provides minimal security. Use MD5 authentication if security is required.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 address** *ipv6-address* / *prefix-length* | *prefix-name ipv6-prefix* / *prefix-length* | **autoconfig** [*default-route*]}
5. **glbp** *group-number* **authentication text** *string*
6. **glbp** *group* **ipv6** [*ipv6-address* | **autoconfig**
7. Repeat Steps 1 through 6 on each router that will communicate.
8. **end**
9. **show glbp**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface Ethernet 0/1`	Configures an interface type and enters interface configuration mode.
Step 4	**ipv6 address** *ipv6-address* / *prefix-length* | *prefix-name ipv6-prefix* / *prefix-length* | **autoconfig** [*default-route*]} **Example:** `Router(config-if)# ipv6 address 2001:DB8:0:7272::72/64`	Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.
Step 5	**glbp** *group-number* **authentication text** *string* **Example:** `Router(config-if)# glbp 10 authentication text stringxyz`	Authenticates GLBP packets received from other routers in the group. • If you configure authentication, all routers within the GLBP group must use the same authentication string.
Step 6	**glbp** *group* **ipv6** [*ipv6-address* | **autoconfig** **Example:** `Router(config-if)# glbp 1 ipv6 FE80::60:3E47:AC8:8`	Enables GLBP in IPv6.
Step 7	Repeat Steps 1 through 6 on each router that will communicate.	--
Step 8	**end** **Example:** `Router(config-if)# end`	Returns to privileged EXEC mode.

	Command or Action	Purpose
Step 9	**show glbp** **Example:** `Router# show glbp`	(Optional) Displays GLBP information. • Use this command to verify your configuration.

Configuring GLBP Weighting Values and Object Tracking

GLBP weighting is used to determine whether a router can act as a virtual forwarder. Initial weighting values can be set and optional thresholds specified. Interface states can be tracked and a decrement value set to reduce the weighting value if the interface goes down. When the GLBP router weighting drops below a specified value, the router will no longer be an active virtual forwarder. When the weighting rises above a specified value, the router can resume its role as an active virtual forwarder.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **track** *object-number* **interface** *type number* {**line-protocol** | **ip routing**
4. **interface** *type number*
5. **glbp** *group* **weighting** *maximum* **lower** *lower*] [**upper** *upper*
6. **glbp** *group* **weighting track** *object-number* [**decrement** *value*]
7. **glbp** *group* **forwarder preempt** [**delay minimum** *seconds*]
8. **end**
9. **show track** [*object-number*| **brief**] [**interface** [**brief**]| **ip route** [**brief**] | **resolution**| **timers**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Command or Action	Purpose
Step 3 **track** *object-number* **interface** *type number* {**line-protocol** | **ip routing** **Example:** `Device(config)# track 2 interface POS 6/0 ip routing`	Configures an interface to be tracked where changes in the state of the interface affect the weighting of a GLBP gateway, and enters tracking configuration mode. • This command configures the interface and corresponding object number to be used with the **glbp weighting track** command. • The **line-protocol** keyword tracks whether the interface is up. The **ip routing** keywords also check that IPv6 routing is enabled on the interface and an IPv6 address is configured.
Step 4 **interface** *type number* **Example:** `Device(config)# interface fastethernet 0/0`	Enters interface configuration mode.
Step 5 **glbp** *group* **weighting** *maximum* **lower** *lower*] [**upper** *upper* **Example:** `Device(config-if)# glbp 10 weighting 110 lower 95 upper 105`	Specifies the initial weighting value, and the upper and lower thresholds, for a GLBP gateway.
Step 6 **glbp** *group* **weighting track** *object-number* [**decrement** *value*] **Example:** `Device(config-if)# glbp 10 weighting track 2 decrement 5`	Specifies an object to be tracked that affects the weighting of a GLBP gateway. • The *value* argument specifies a reduction in the weighting of a GLBP gateway when a tracked object fails.
Step 7 **glbp** *group* **forwarder preempt** [**delay minimum** *seconds*] **Example:** `Device(config-if)# glbp 10 forwarder preempt delay minimum 60`	Configures the router to take over as the AVF for a GLBP group if the current AVF for a GLBP group falls below its low weighting threshold. • This command is enabled by default with a delay of 30 seconds. • Use the optional **delay** and **minimum** keywords and the *seconds* argument to specify a minimum delay interval in seconds before preemption of the AVF takes place.
Step 8 **end** **Example:** `Device(config-if)# exit`	Returns to privileged EXEC mode.

	Command or Action	Purpose
Step 9	**show track** [*object-number*| **brief**] [**interface** [**brief**]| **ip route** [**brief**]	**resolution**| **timers**] **Example:** `Device# show track 2`	Displays tracking information.

Enabling and Verifying GLBP

GLBP is designed to be easy to configure. Each gateway in a GLBP group must be configured with the same group number, and at least one gateway in the GLBP group must be configured with the virtual IPv6 address to be used by the group. All other required parameters can be learned.

If VLANs are in use on an interface, the GLBP group number must be different for each VLAN.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 address** *ipv6-address* / *prefix-length* | *prefix-name ipv6-prefix* / *prefix-length* | **autoconfig** [*default-route*]}
5. **glbp** *group* **ipv6** [*ipv6-address* | **autoconfig**]
6. **exit**
7. **show glbp** [*interface-type interface-number*] [*group*] [*state*] [**brief**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Device(config)# interface fastethernet 0/0`	Specifies an interface type and number, and enters interface configuration mode.
Step 4	**ipv6 address** *ipv6-address / prefix-length | prefix-name ipv6-prefix / prefix-length* | **autoconfig** [*default-route*]} **Example:** `Device(config-if)# ipv6 address 2001:DB8:0:7262::62/64`	Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.
Step 5	**glbp** *group* **ipv6** [*ipv6-address* | **autoconfig**] **Example:** `Device(config-if)# glbp 1 ipv6 FE80::60:3E47:AC8:8`	Enables GLBP in IPv6.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode, and returns the router to global configuration mode.
Step 7	**show glbp** [*interface-type interface-number*] [*group*] [*state*] [**brief**] **Example:** `Device(config)# show glbp 10`	(Optional) Displays information about GLBP groups on a router. • Use the optional **brief** keyword to display a single line of information about each virtual gateway or virtual forwarder.

Troubleshooting GLBP

This task requires a router running GLBP to be attached directly to a console.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no logging console**
4. Use Telnet to access a router port and repeat Steps 1 and 2.
5. **end**
6. **terminal monitor**
7. **debug condition glbp** *interface-type interface-number group* [*forwarder*]
8. **terminal no monitor**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no logging console** **Example:** `Device(config)#` **`no logging console`**	Disables all logging to the console terminal. • To reenable logging to the console, use the **logging console** command in global configuration mode.
Step 4	Use Telnet to access a router port and repeat Steps 1 and 2.	Enters global configuration mode in a recursive Telnet session, which allows the output to be redirected away from the console port.
Step 5	**end** **Example:** `Device(config)# end`	Exits to privileged EXEC mode.

	Command or Action	Purpose
Step 6	**terminal monitor** **Example:** `Device#` **terminal monitor**	Enables logging output on the virtual terminal.
Step 7	**debug condition glbp** *interface-type interface-number group* [*forwarder*] **Example:** `Device#` **debug condition glbp fastethernet 0/0 10 1**	Displays debugging messages about GLBP conditions. • Try to enter only specific **debug condition glbp**or **debug glbp**commands to isolate the output to a certain subcomponent and minimize the load on the processor. Use appropriate arguments and keywords to generate more detailed debug information on specified subcomponents. • Enter the specific **no debug condition glbp**or **no debug glbp** command when you are finished.
Step 8	**terminal no monitor** **Example:** `Device#` **terminal no monitor**	Disables logging on the virtual terminal.

Enabling an HSRP Group for IPv6 Operation

HSRP version 2 must be enabled on an interface before HSRP IPv6 can be configured.

- Enabling HSRP Version 2, page 308
- Enabling and Verifying an HSRP Group for IPv6 Operation, page 309

Enabling HSRP Version 2

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **standby version** {**1**| **2**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface ethernet 0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**standby version** {**1**| **2** **Example:** `Router(config-if)# standby version 2`	Changes the version of the HSRP. • Version 1 is the default.

Enabling and Verifying an HSRP Group for IPv6 Operation

In this task, when you enter the **standby ipv6** command, a modified EUI-64 format interface identifier is generated in which the EUI-64 interface identifier is created from the relevant HSRP virtual MAC address.

In IPv6, a router on the link advertises in RA messages any site-local and global prefixes, and its willingness to function as a default router for the link. RA messages are sent periodically and in response to router solicitation messages, which are sent by hosts at system startup.

A node on the link can automatically configure site-local and global IPv6 addresses by appending its interface identifier (64 bits) to the prefixes (64 bits) included in the RA messages. The resulting 128-bit IPv6 addresses configured by the node are then subjected to duplicate address detection to ensure their uniqueness on the link. If the prefixes advertised in the RA messages are globally unique, then the IPv6 addresses configured by the node are also guaranteed to be globally unique. Router solicitation messages, which have a value of 133 in the Type field of the ICMPv6 packet header, are sent by hosts at system startup so that the host can immediately autoconfigure without needing to wait for the next scheduled RA message.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **interface** *type number*
5. **standby** [*group-number*] **ipv6** {*ipv6-global-address* | *ipv6-address* / *prefix-length* | *ipv6-prefix* / *prefix-length* | *link-local-address* | **autoconfig**
6. **standby** [*group-number*] **preempt** [**delay minimum** *seconds* | **reload** *seconds* | **sync** *seconds*}]
7. **standby** [*group-number*] **priority** *priority*
8. **exit**
9. **show standby** [*type number* [*group*]] [**all** | **brief**
10. **show ipv6 interface** [**brief**] [*interface-type interface-number*] [**prefix**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Router(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams. • The **ipv6 unicast-routing** command must be enabled for HSRP for IPv6 to work.
Step 4	**interface** *type number* **Example:** `Router(config)# interface ethernet 0/0`	Specifies an interface type and number, and places the router in interface configuration mode.

	Command or Action	Purpose
Step 5	**standby** [*group-number*] **ipv6** {*ipv6-global-address* | *ipv6-address* / *prefix-length* | *ipv6-prefix* / *prefix-length* | *link-local-address* | **autoconfig** **Example:** `Router(config-if)# standby 1 ipv6 autoconfig`	Activates the HSRP in IPv6. If an IPv6 address is used, it must include an IPv6 prefix length. If a link-local address is used, it must not have a prefix.
Step 6	**standby** [*group-number*] **preempt** [**delay minimum** *seconds* | **reload** *seconds* | **sync** *seconds*}] **Example:** `Router(config-if)# standby 1 preempt`	Configures HSRP preemption and preemption delay.
Step 7	**standby** [*group-number*] **priority** *priority* **Example:** `Router(config-if)# standby 1 priority 110`	Configures HSRP priority.
Step 8	**exit** **Example:** `Router(config-if)# exit`	Returns the router to privileged EXEC mode.
Step 9	**show standby** [*type number* [*group*]] [**all** | **brief** **Example:** `Router# show standby`	Displays HSRP information.
Step 10	**show ipv6 interface** [**brief**] [*interface-type interface-number*] [**prefix** **Example:** `Router# show ipv6 interface ethernet 0/0`	Displays the usability status of interfaces configured for IPv6.

Configuration Examples for First Hop Redundancy Protocols in IPv6

Example: Customizing GLBP Configuration

In the following example, Router A, shown in Figure 1, is configured with a number of GLBP commands:

```
interface fastethernet 0/0
 ipv6 address 2001:DB8:0001:0001:/64
 glbp 10 timers 5 18
 glbp 10 timers redirect 600 7200
 glbp 10 load-balancing host-dependent
 glbp 10 priority 254
 glbp 10 preempt delay minimum 60
```

Example GLBP MD5 Authentication Using Key Strings

The following example configures GLBP MD5 authentication using a key string:

```
!
interface Ethernet 0/1
 ipv6 address 2001:DB8:0001:0001:/64
 glbp 2 authentication md5 key-string ThisStringIsTheSecretKey
 glbp 2 ipv6 FE80::260:3EFF:FE11:6770
```

Example GLBP MD5 Authentication Using Key Chains

In the following example, GLBP queries the key chain "AuthenticateGLBP" to obtain the current live key and key ID for the specified key chain:

```
key chain AuthenticateGLBP
 key 1
  key-string ThisIsASecretKey
interface Ethernet 0/1
 ipv6 address 2001:DB8:0001:0001:/64
 glbp 2 authentication md5 key-chain AuthenticateGLBP
 glbp 2 ipv6 FE80::E0:F727:E400:A
```

Example GLBP Text Authentication

The following example configures GLBP text authentication using a text string:

```
interface fastethernet 0/0
 ipv6 address 2001:DB8:0001:0001:/64
 glbp 10 authentication text stringxyz
 glbp 10 ipv6 FE80::60:3E47:AC8:8
```

Example: GLBP Weighting

In the following example, Router A, shown in Figure 1, is configured to track the IP routing state of the POS interfaces 5/0 and 6/0, an initial GLBP weighting with upper and lower thresholds is set, and a

weighting decrement value of 10 is set. If POS interfaces 5/0 and 6/0 go down, the weighting value of the router is reduced.

```
track 1 interface POS 5/0 ip routing
track 2 interface POS 6/0 ip routing
interface fastethernet 0/0
 glbp 10 weighting 110 lower 95 upper 105
 glbp 10 weighting track 1 decrement 10
 glbp 10 weighting track 2 decrement 10
 glbp 10 forwarder preempt delay minimum 60
```

Example: Enabling GLBP Configuration

In the following example, the router is configured to enable GLBP, and the virtual IPv6 address of 2001:DB8:0002:0002:/64 is specified for GLBP group 10:

```
interface fastethernet 0/0
ipv6 address 2001:DB8:0001:0001:/64
 glbp 10 ipv6 FE80::60:3E47:AC8:8
```

In the following example, GLBP for IPv6 is enabled for GLBP group 15:

```
interface fastethernet 0/0
 ipv6 address 2001:DB8:0001:0001:/64
 glbp 10 ipv6
```

Example Enabling and Verifying an HSRP Group for IPv6 Operation

Example: Configuration and Verification for an HSRP Group

The following example shows configuration and verification for an HSRP group for IPv6 that consists of Router1 and Router2. The **show standby** command is issued for each router to verify the router's configuration.

Router 1 Configuration

```
interface FastEthernet0/0.100
description DATA VLAN for PCs
encapsulation dot1Q 100
ipv6 address 2001:DB8:CAFE:2100::BAD1:1010/64
standby version 2
standby 101 priority 120
standby 101 preempt delay minimum 30
standby 101 authentication ese
standby 101 track Serial0/1/0.17 90
standby 201 ipv6 autoconfig
standby 201 priority 120
standby 201 preempt delay minimum 30
standby 201 authentication ese
standby 201 track Serial0/1/0.17 90
Router1# show standby
FastEthernet0/0.100 - Group 101 (version 2)
State is Active
2 state changes, last state change 5w5d
Active virtual MAC address is 0000.0c9f.f065
Local virtual MAC address is 0000.0c9f.f065 (v2 default)
Hello time 3 sec, hold time 10 sec
```

```
Next hello sent in 2.296 secs
Authentication text "ese"
Preemption enabled, delay min 30 secs
Active router is local
Priority 120 (configured 120)
Track interface Serial0/1/0.17 state Up decrement 90
IP redundancy name is "hsrp-Fa0/0.100-101" (default)
FastEthernet0/0.100 - Group 201 (version 2)
State is Active
2 state changes, last state change 5w5d
Virtual IP address is FE80::5:73FF:FEA0:C9
Active virtual MAC address is 0005.73a0.00c9
Local virtual MAC address is 0005.73a0.00c9 (v2 IPv6 default)
Hello time 3 sec, hold time 10 sec
Next hello sent in 2.428 secs
Authentication text "ese"
Preemption enabled, delay min 30 secs
Active router is local
Standby router is FE80::20F:8FFF:FE37:3B70, priority 100 (expires in 7.856 sec)
Priority 120 (configured 120)
Track interface Serial0/1/0.17 state Up decrement 90
IP redundancy name is "hsrp-Fa0/0.100-201" (default)
```

Router 2 Configuration

```
interface FastEthernet0/0.100
description DATA VLAN for Computers
encapsulation dot1Q 100
ipv6 address 2001:DB8:CAFE:2100::BAD1:1020/64
standby version 2
standby 101 preempt
standby 101 authentication ese
standby 201 ipv6 autoconfig
standby 201 preempt
standby 201 authentication ese
Router2# show standby
FastEthernet0/0.100 - Group 101 (version 2)
State is Standby
7 state changes, last state change 5w5d
Active virtual MAC address is 0000.0c9f.f065
Local virtual MAC address is 0000.0c9f.f065 (v2 default)
Hello time 3 sec, hold time 10 sec
Next hello sent in 0.936 secs
Authentication text "ese"
Preemption enabled
MAC address is 0012.7fc6.8f0c
Standby router is local
Priority 100 (default 100)
IP redundancy name is "hsrp-Fa0/0.100-101" (default)
FastEthernet0/0.100 - Group 201 (version 2)
State is Standby
7 state changes, last state change 5w5d
Virtual IP address is FE80::5:73FF:FEA0:C9
Active virtual MAC address is 0005.73a0.00c9
Local virtual MAC address is 0005.73a0.00c9 (v2 IPv6 default)
Hello time 3 sec, hold time 10 sec
Next hello sent in 0.936 secs
Authentication text "ese"
Preemption enabled
Active router is FE80::212:7FFF:FEC6:8F0C, priority 120 (expires in 7.548 sec)
MAC address is 0012.7fc6.8f0c
Standby router is local
Priority 100 (default 100)
IP redundancy name is "hsrp-Fa0/0.100-201" (default)
```

Example: Configuring HSRP Global IPv6 Addresses

This example shows three HSRP global IPv6 addresses with an explicitly configured link-local address:

```
interface Ethernet0/0
```

```
 no ip address
 ipv6 address 2001::DB8:1/64
 standby version 2
 standby 1 ipv6 FE80::1:CAFÉ
 standby 1 ipv6 2001::DB8:2/64
 standby 1 ipv6 2001:DB8::3/64
 standby 1 ipv6 2001:DB8::4/64
end
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 link-local addresses and stateless autoconfiguration	" Implementing IPv6 Addressing and Basic Connectivity ," *Cisco IOS IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*
Configuring HSRP in IPv4	" Configuring HSRP ," *Cisco IOS IP Application Services Configuration Guide*

Standards

Standard	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2281	*Cisco Hot Standby Router Protocol (HSRP)*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for First Hop Redundancy Protocols in IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 16 Feature Information for First Hop Redundancy Protocols for IPv6

Feature Name	Releases	Feature Configuration Information
FHRP--GLBP Support for IPv6	12.2(58)SE 12.2(33)SXI 12.4(6)T	GLBP protects data traffic from a failed router or circuit while allowing packet load sharing between a group of redundant routers. The following commands were introduced or modified for this feature: **glbp forwarder preempt**, **glbp ipv6**, **glbp load-balancing**, **glbp preempt**, **glbp priority**, **glbp name**, **glbp timers**, **glbp timers redirect**, **glbp weighting**, **glbp weighting track**, **track interface**.

Feature Name	Releases	Feature Configuration Information
GLBP MD5 Authentication	12.2(18)S 12.3(2)T	MD5 authentication provides greater security than the alternative plain text authentication scheme. MD5 authentication allows each GLBP group member to use a secret key to generate a keyed MD5 hash that is part of the outgoing packet. A keyed hash of an incoming packet is generated and, if the hash within the incoming packet does not match the generated hash, the packet is ignored. The following commands were introduced or modified for this feature: **glbp authentication**, **key**, **key chain**, **key-string (authentication)**, **show glbp**, **show key chain**.
IPv6 Services--HSRP for IPv6	12.4(4)T 1 2.2(33)SRB 12.2(33)SXI	The HSRP is an FHRP designed to allow for transparent failover of the first-hop IPv6 router. The following commands were introduced or modified by this feature: **show standby**, standby ipv6, standby preempt, standby priority.
HSRP--Global IPv6 Addresses	12.2(33)SXI4 15.0(1)SY 15.1(1)SG	The HSRP global IPv6 address feature allows users to configure multiple non-link local addresses as virtual addresses. The following command was modified by this feature: standby ipv6.

Glossary

- **CPE** --Customer premises equipment
- **FHRP** --First hop redundancy protocol
- **GLBP** --Gateway load balancing protocol
- **HSRP** --Hot standby routing protocol
- **NA** --Neighbor advertisement
- **ND** --Neighbor Discovery
- **NS** --Neighbor solicitation
- **PE** --Provider equipment
- **RA** --Router advertisement
- **RS** --Router solicitation

Implementing First Hop Security in IPv6

This document provides information about configuring features that comprise first hop security functionality in IPv6.

A database table of IPv6 neighbors connected to the switch is created from information sources such as Neighbor Discovery (ND) protocol snooping. This database, or binding, table is used by various IPv6 guard features (such as IPv6 ND Inspection, per-port address limit, IPv6 device tracking) to validate the link-layer address (LLA), the IPv4 or IPv6 address, and prefix binding of the neighbors to prevent spoofing and redirect attacks.

IPv6 ND Inspection learns and secures bindings for stateless autoconfiguration addresses in layer 2 neighbor tables. IPv6 ND inspection analyzes neighbor discovery messages in order to build a trusted binding table database, and IPv6 neighbor discovery messages that do not conform are dropped.

Router advertisements (RAs) are used by routers to announce themselves on the link. IPv6 RA Guard analyzes these RAs and can filter out bogus ones sent by unauthorized routers.

The per-port address limit feature enables an operator to specify a maximum number of IPv6 addresses allowed on a port of the switch. This function is achieved by filtering out ND messages sourced with addresses beyond the per-port address limit.

IPv6 Device Tracking provides IPv6 host liveness tracking so that a neighbor table can be immediately updated when an IPv6 host disappears.

The Secure Neighbor Discovery for Cisco IOS Software feature is designed to counter the threats of the ND protocol. Secure neighbor discovery (SeND) defines a set of neighbor discovery options and two neighbor discovery messages. SeND also defines a new autoconfiguration mechanism to establish address ownership. The IPv6 PACL feature adds IPv6 port-based ACL support.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release.

To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Implementing First Hop Security in IPv6

- You should be familiar with the IPv6 neighbor discovery feature. For information about IPv6 neighbor discovery, see "Implementing IPv6 Addressing and Basic Connectivity".
- The SeND feature is available on crypto images because it involves using cryptographic libraries.
- In order to use IPv6 port-based access list (PACL), you must know how to configure IPv6 access lists. For information about configuring IPv6 access lists, see "Implementing Traffic Filters and Firewalls for IPv6 Security".

Restrictions for Implementing First Hop Security in IPv6

The IPv6 PACL feature is supported only in the ingress direction; it is not supported in the egress direction.

RA Guard in Cisco IOS Release 12.2(33)SXI4

- The RA guard feature does not offer protection in environments where IPv6 traffic is tunneled.
- This feature is supported only in hardware by programming the TCAM.
- This feature can be configured only on a switchport interface in the ingress direction.
- This feature supports only host mode.
- This feature is supported only in the ingress direction; it is not supported in the egress direction.
- This feature is supported on ether channel, but not on ether channel port members.
- This feature is not supported on trunk ports with merge mode.
- This feature is supported on auxiliary VLANs and PVLANs. In case of PVLANs, primary VLAN features are inherited and merged with port features.
- Packets dropped by the RA guard feature can be spanned.
- If the **platform ipv6 acl icmp optimize neighbor-discovery** command is configured, RA guard feature configuration should not be allowed and an error message should be displayed. This command adds default global ICMP entries that will override the RA guard ICMP entries.

Information About Implementing First Hop Security in IPv6

IPv6 First-Hop Security Binding Table

A database table of IPv6 neighbors connected to the device is created from information sources such as Neighbor Discovery (ND) protocol snooping. This database, or binding, table is used by various IPv6 guard features to validate the link-layer address (LLA), the IPv4 or IPv6 address, and prefix binding of the neighbors to prevent spoofing and redirect attacks.

IPv6 Device Tracking

The IPv6 device tracking feature provides IPv6 host liveness tracking so that a neighbor table can be immediately updated when an IPv6 host disappears. The feature tracks the liveness of the neighbors connected through the Layer 2 device on regular basis in order to revoke network access privileges as they become inactive.

IPv6 Port-Based Access Control List Support

The IPv6 PACL feature provides the ability to provide access control (permit or deny) on Layer 2 switch ports for IPv6 traffic. IPv6 PACLs are similar to IPv4 PACLs, which provide access control on Layer 2 switch ports for IPv4 traffic. They are supported only in the ingress direction and in hardware.

A PACL can filter ingress traffic on Layer 2 interfaces based on Layer 3 and Layer 4 header information or non-IP Layer 2 information.

IPv6 Global Policies

IPv6 global policies provide storage and access policy database services. IPv6 neighbor discovery (ND) inspection and the IPv6 Router Advertisement (RA) Guard are IPv6 global policies features. Every time an ND inspection or RA guard is configured globally, the policy attributes are stored in the software policy database. The policy is then applied to an interface, and the software policy database entry is updated to include this interface to which the policy is applied.

- IPv6 RA Guard, page 321
- IPv6 ND Inspection, page 321

IPv6 RA Guard

The IPv6 RA Guard feature provides support for allowing the network administrator to block or reject unwanted or rogue router advertisement (RA) guard messages that arrive at the network device platform. RAs are used by devices to announce themselves on the link. The IPv6 RA Guard feature analyzes these RAs and filters out RAs that are sent by unauthorized devices. In host mode, all RA and router redirect messages are disallowed on the port. The RA guard feature compares configuration information on the Layer 2 (L2) device with the information found in the received RA frame. Once the L2 device has validated the content of the RA frame and router redirect frame against the configuration, it forwards the RA to its unicast or multicast destination. If the RA frame content is not validated, the RA is dropped.

IPv6 ND Inspection

IPv6 ND inspection learns and secures bindings for stateless autoconfiguration addresses in Layer 2 neighbor tables. IPv6 ND inspection analyzes neighbor discovery messages in order to build a trusted binding table database, and IPv6 neighbor discovery messages that do not have valid bindings are dropped. A neighbor discovery message is considered trustworthy if its IPv6-to-MAC mapping is verifiable.

This feature mitigates some of the inherent vulnerabilities for the neighbor discovery mechanism, such as attacks on duplicate address detection (DAD), address resolution, device discovery, and the neighbor cache.

Secure Neighbor Discovery in IPv6

IPv6 Neighbor Discovery Trust Models and Threats

There are three IPv6 neighbor discovery trust models, which are described as follows:

- All authenticated nodes trust each other to behave correctly at the IP layer and not to send any neighbor discovery or router discovery (RD) messages that contain false information. This model represents a situation where the nodes are under a single administration and form a closed or semiclosed group. A corporate intranet is an example of this model.
- A router trusted by the other nodes in the network to be a legitimate router that routes packets between the local network and any connected external networks. This router is trusted to behave correctly at the IP layer and not to send any neighbor discovery or RD messages that contain false information. This model represents a public network run by an operator. The clients pay the operator, have the operator's credentials, and trust the operator to provide the IP forwarding service. The clients do not trust each other to behave correctly; any other client node must be considered able to send falsified neighbor discovery and RD messages.
- A model where the nodes do not directly trust each other at the IP layer. This model is considered suitable where a trusted network operator is not available.

Nodes on the same link use ND to discover each other's presence and link-layer addresses, to find routers, and to maintain reachability information about the paths to active neighbors. ND is used by both hosts and routers. The original ND specifications used IPsec to protect ND messages. However, not many detailed instructions for using IPsec are available. The number of manually configured security associations needed for protecting ND can be very large, which makes that approach impractical for most purposes. These threats need to be considered and eliminated.

SeND Protocol

The SeND protocol counters ND threats. It defines a set of new ND options, and two new ND messages (Certification Path Solicitation [CPS] and Certification Path Answer [CPA]). It also defines a new autoconfiguration mechanism to be used in conjunction with the new ND options to establish address ownership.

SeND defines the mechanisms defined in the following sections for securing ND:

Cryptographically Generated Addresses in SeND

Cryptographically generated addresses (CGAs) are IPv6 addresses generated from the cryptographic hash of a public key and auxiliary parameters. This provides a method for securely associating a cryptographic public key with an IPv6 address in the SeND protocol.

The node generating a CGA address must first obtain a Rivest, Shamir, and Adelman (RSA) key pair (SeND uses an RSA public/private key pair). The node then computes the interface identifier part (which is the rightmost 64 bits) and appends the result to the prefix to form the CGA address.

CGA address generation is a one-time event. A valid CGA cannot be spoofed and the CGA parameters received associated to it is reused because the message must be signed with the private key that matches the public key used for CGA generation, which only the address owner will have.

A user cannot replay the complete SeND message (including the CGA address, CGA parameters, and CGA signature) because the signature has only a limited lifetime.

Authorization Delegation Discovery

Authorization delegation discovery is used to certify the authority of devices by using a trust anchor. A trust anchor is a third party that the host trusts and to which the device has a certification path. At a basic level, the device is certified by the trust anchor. In a more complex environment, the device is certified by a user that is certified by the trust anchor. In addition to certifying the device identity (or the right for a node to act as a device), the certification path contains information about prefixes that a device is allowed to advertise in RAs. Authorization delegation discovery enables a node to adopt a device as its default device.

SeND Deployment Models

Host-to-Host Deployment Without a Trust Anchor

Deployment for SeND between hosts is straightforward. The hosts can generate a pair of RSA keys locally, autoconfigure their CGA addresses, and use them to validate their sender authority, rather than using a trust anchor to establish sender authority. The figure below illustrates this model.

Figure 25 Host-to-Host Deployment Model

Neighbor Solicitation Flow

In a neighbor solicitation scenario, hosts and devices in host mode exchange neighbor solicitations and neighbor advertisements. These neighbor solicitations and neighbor advertisements are secured with CGA addresses and CGA options, and have nonce, time stamp, and RSA neighbor discovery options. The figure below illustrates this scenario.

Figure 26 Neighbor Solicitation Flow

Host-Router Deployment Model

In many cases, hosts will not have access to the infrastructure that will enable them to obtain and announce their certificates. In these situations, hosts will secure their relationship using CGA, and secure their relationship with routers using a trusted anchor. When using RAs, SeND mandates that routers are authenticated through a trust anchor. The figure below illustrates this scenario.

Figure 27 Host-Router Deployment Model

Router Advertisement and Certificate Path Flows

The figure below shows the certificate exchange performed using certification path solicitation CPS/CPA SeND messages. In the illustration, Router R is certified (using an X.509 certificate) by its own CA (certificates CR). The CA itself (CA2) is certified by its own CA (certificates C2), and so on, up to a CA (CA0) that the hosts trusts. The certificate CR contains IP extensions per RFC 3779, which describes which prefix ranges the router R is allowed to announce (in RAs). This prefix range, certified by CA2, is a subset of CA2's own range, certified by CA1, and so on. Part of the validation process when a certification chain is received consists of validating the certification chain and the consistency of nested prefix ranges.

Figure 28 Router Advertisement and Certificate Path Flows

Single CA Model

The deployment model shown in the third figure above can be simplified in an environment where both hosts and routers trust a single CA such as the Cisco certification server (CS). The figure below illustrates this model.

Figure 29 Single CA Deployment Model

How to Implement First Hop Security in IPv6

- Configuring the IPv6 Binding Table Content, page 325
- Configuring IPv6 Device Tracking, page 326
- Configuring IPv6 ND Inspection, page 327
- Configuring IPv6 RA Guard, page 331
- Configuring SeND for IPv6, page 334
- Configuring IPv6 PACL, page 358

Configuring the IPv6 Binding Table Content

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 neighbor binding vlan** *vlan-id* {**interface** *type number* | *ipv6-address* | *mac-address*} [**tracking** [**disable** | **enable** | **retry-interval** *value*] | **reachable-lifetime** *value*]
4. **ipv6 neighbor binding max-entries** *entries* [**vlan-limit** *number* | **interface-limit** *number* | **mac-limit** *number*]
5. **ipv6 neighbor binding logging**
6. **exit**
7. **show ipv6 neighbor binding** [**vlan** *vlan-id* | **interface** *type number* | **ipv6** *ipv6-address* | **mac** *mac-address*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 neighbor binding vlan** *vlan-id* {**interface** *type number*	*ipv6-address*	*mac-address*} [**tracking** [**disable**	**enable**	**retry-interval** *value*]	**reachable-lifetime** *value*] **Example:** `Device(config)# ipv6 neighbor binding vlan 100 interface Ethernet 0/0 reachable-lifetime 100`	Adds a static entry to the binding table database.
Step 4	**ipv6 neighbor binding max-entries** *entries* [**vlan-limit** *number*	**interface-limit** *number*	**mac-limit** *number*] **Example:** `Device(config)# ipv6 neighbor binding max-entries 100`	Specifies the maximum number of entries that are allowed to be inserted in the binding table cache.

	Command or Action	Purpose
Step 5	**ipv6 neighbor binding logging** **Example:** `Device(config)# ipv6 neighbor binding logging`	Enables the logging of binding table main events.
Step 6	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode, and places the device in privileged EXEC mode.
Step 7	**show ipv6 neighbor binding** [**vlan** *vlan-id* | **interface** *type number* | **ipv6** *ipv6-address* | **mac** *mac-address*] **Example:** `Device# show ipv6 neighbor binding`	Displays the contents of a binding table.

Configuring IPv6 Device Tracking

Perform this task to provide fine tuning for the life cycle of an entry in the binding table for the IPv6 device tracking feature For IPv6 device tracking to work, the binding table needs to be populated.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 neighbor tracking** [**retry-interval** *value*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ipv6 neighbor tracking** [**retry-interval** *value*] **Example:** `Device(config)# ipv6 neighbor tracking`	Tracks entries in the binding table.

Configuring IPv6 ND Inspection

Configuring IPv6 ND Inspection Globally

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 nd inspection policy** *policy-name*
4. **drop-unsecure**
5. **sec-level minimum** *value*
6. **device-role** {**host** | **monitor** | **router**}
7. **tracking** {**enable** [**reachable-lifetime** {*value* | **infinite**}] | **disable** [**stale-lifetime** {*value* | **infinite**}]}
8. **trusted-port**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ipv6 nd inspection policy** *policy-name* **Example:** `Device(config)# ipv6 nd inspection policy policy1`	Defines the ND inspection policy name and places the device in ND inspection policy configuration mode.
Step 4	**drop-unsecure** **Example:** `Device(config-nd-inspection)# drop-unsecure`	Drops messages with no options, invalid options, or an invalid signature.
Step 5	**sec-level minimum** *value* **Example:** `Device(config-nd-inspection)# sec-level minimum 2`	Specifies the minimum security level parameter value when cryptographically generated address (CGA) options are used.
Step 6	**device-role** {**host** | **monitor** | **router**} **Example:** `Device(config-nd-inspection)# device-role monitor`	Specifies the role of the device attached to the port.
Step 7	**tracking** {**enable** [**reachable-lifetime** {*value* | **infinite**}] | **disable** [**stale-lifetime** {*value* | **infinite**}]} **Example:** `Device(config-nd-inspection)# tracking disable stale-lifetime infinite`	Overrides the default tracking policy on a port.
Step 8	**trusted-port** **Example:** `Device(config-nd-inspection)# trusted-port`	Configures a port to become a trusted port.

Applying IPv6 ND Inspection on a Specified Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 nd inspection** [**attach-policy** [**policy** *policy-name*] | **vlan** {**add** | **except** | **none** | **remove** | **all**} *vlan* [*vlan1, vlan2, vlan3...*]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface fastethernet 0/0`	Specifies an interface type and number, and places the device in interface configuration mode.
Step 4	**ipv6 nd inspection** [**attach-policy** [**policy** *policy-name*] | **vlan** {**add** | **except** | **none** | **remove** | **all**} *vlan* [*vlan1, vlan2, vlan3...*]] **Example:** `Device(config-if)# ipv6 nd inspection`	Applies the ND inspection feature on the interface.

Verifying and Troubleshooting IPv6 ND Inspection

These optional commands can be entered in any order to verify and troubleshoot the IPv6 ND inspection feature.

SUMMARY STEPS

1. **enable**
2. **show ipv6 snooping capture-policy** [**interface** *type number*]
3. **show ipv6 snooping counter** [**interface** *type number*]
4. **show ipv6 snooping features**
5. **show ipv6 snooping policies** [**interface** *type number*]
6. **debug ipv6 snooping** [**binding-table** | **classifier** | **errors** | **feature-manager** | **filter** *acl* | **ha** | **hw-api** | **interface** *interface* | **memory** | **ndp-inspection** | **policy** | **vlan** *vlanid* | **switcher** | **filter** *acl* | **interface** *interface* | *vlanid*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ipv6 snooping capture-policy** [**interface** *type number*] **Example:** `Device# show ipv6 snooping capture-policy interface ethernet 0/0`	Displays snooping ND message capture policies.
Step 3	**show ipv6 snooping counter** [**interface** *type number*] **Example:** `Device# show ipv6 snooping counter interface Fa 4/12`	Displays information about the packets counted by the interface counter.
Step 4	**show ipv6 snooping features** **Example:** `Device# show ipv6 snooping features`	Displays information about snooping features configured on the device.
Step 5	**show ipv6 snooping policies** [**interface** *type number*] **Example:** `Device# show ipv6 snooping policies`	Displays information about the configured policies and the interfaces to which they are attached.

	Command or Action	Purpose
Step 6	**debug ipv6 snooping** [**binding-table** | **classifier** | **errors** | **feature-manager** | **filter** *acl* | **ha** | **hw-api** | **interface** *interface* | **memory** | **ndp-inspection** | **policy** | **vlan** *vlanid* | **switcher** | **filter** *acl* | **interface** *interface* | *vlanid*] **Example:** `Device# debug ipv6 snooping`	Enables debugging for snooping information in IPv6.

Configuring IPv6 RA Guard

Configuring the IPv6 RA Guard on a Specified Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 nd raguard attach-policy** [*policy-name* [**vlan** {**add** | **except** | **none** | **remove** | **all**} *vlan* [*vlan1*, *vlan2*, *vlan3*...]]]
5. **exit**
6. **show ipv6 nd raguard policy** [*policy-name*]
7. **debug ipv6 snooping raguard** [*filter* | *interface* | *vlanid*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Device(config)# interface fastethernet 3/13`	Specifies an interface type and number, and places the device in interface configuration mode.
Step 4	**ipv6 nd raguard attach-policy** [*policy-name* [**vlan** {**add** | **except** | **none** | **remove** | **all**} *vlan* [*vlan1, vlan2, vlan3...*]]] **Example:** `Device(config-if)# ipv6 nd raguard attach-policy`	Applies the IPv6 Router Advertisement (RA) guard feature to a specified interface.
Step 5	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.
Step 6	**show ipv6 nd raguard policy** [*policy-name*] **Example:** `Device# show ipv6 nd raguard policy raguard1`	Displays the RA guard policy on all interfaces configured with the RA guard.
Step 7	**debug ipv6 snooping raguard** [*filter* | *interface* | *vlanid*] **Example:** `Device# debug ipv6 snooping raguard`	Enables debugging for IPv6 RA guard snooping information.

- Configuring IPv6 RA Guard in Cisco IOS Release 12.2(33)SXI4 and 12.2(54)SG, page 332

Configuring IPv6 RA Guard in Cisco IOS Release 12.2(33)SXI4 and 12.2(54)SG

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 nd raguard**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface fastethernet 0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**ipv6 nd raguard** **Example:** `Router(config-if)# ipv6 nd raguard`	Applies the IPv6 RA guard feature.

Verifying and Troubleshooting IPv6 RA Guard

SUMMARY STEPS

1. **enable**
2. **show ipv6 nd raguard policy** [*policy-name*]
3. **debug ipv6 snooping raguard** [*filter* | *interface* | *vlanid*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**show ipv6 nd raguard policy** [*policy-name*] **Example:** `Router# show ipv6 nd raguard policy raguard1`	Displays RAs guard policy on all interfaces configured with RA guard.
Step 3	**debug ipv6 snooping raguard** [*filter* | *interface* | *vlanid*] **Example:** `Router# debug ipv6 snooping raguard`	Enables debugging for snooping information in the IPv6 RA guard feature

Configuring SeND for IPv6

Certificate servers are used to grant certificates after validating or certifying key pairs. A tool for granting certificates is mandatory in any SeND deployment. Many tools are available to grant certificates, for example, Open Secure Sockets Layer (OpenSSL) on Linux. However, very few certificate servers support granting certificates containing IP extensions. Cisco IOS certificate servers support every kind of certificate including the certificates containing the IP extensions.

SeND is available in host mode. The set of available functions on a host will be a subset of SeND functionality. CGA will be fully available and the prefix authorization delegation will be supported on the host side (sending CPS and receiving CPA).

To implement SeND, configure the host with the following parameters:

- An RSA key pair used to generate CGA addresses on the interface.
- A SeND modifier that is computed using the RSA key pair.
- A key on the SeND interface.
- CGAs on the SeND interface.
- A Public Key Infrastructure (PKI) trustpoint, with minimum content; for example, the URL of the certificate server. A trust anchor certificate must be provisioned on the host.

SeND is also available in router mode. You can use the **ipv6 unicast-routing** command to configure a node to a router. To implement SeND, configure routers with the same elements as that of the host. The routers will need to retrieve certificates of their own from a certificate server. The RSA key and subject name of the trustpoint are used to retrieve certificates from a certificate server. Once the certificate has been obtained and uploaded, the router generates a certificate request to the certificate server and installs the certificate.

The following operations need to be completed before SeND is configured on the host or router:

- Hosts are configured with one or more trust anchors.
- Hosts are configured with an RSA key pair or configured with the capability to locally generate it. Note that for hosts not establishing their own authority via a trust anchor, these keys are not certified by any CA.
- Routers are configured with RSA keys and corresponding certificate chains, or the capability to obtain these certificate chains that match the host trust anchor at some level of the chain.

While booting, hosts and routers must either retrieve or generate their CGAs. Typically, routers will autoconfigure their CGAs once and save them (along with the key pair used in the CGA operation) into

their permanent storage. At a minimum, link-local addresses on a SeND interface should be CGAs. Additionally, global addresses can be CGAs.

Configuring Certificate Servers to Enable SeND

Hosts and routers must be configured with RSA key pairs and corresponding certificate chains before the SeND parameters are configured. Perform the following task to configure the certificate server to grant certificates. Once the certificate server is configured, other parameters for the certificate server can be configured.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip http server**
4. **crypto pki trustpoint** *name*
5. **ip-extension** [**multicast** | **unicast**] {**inherit** [**ipv4** | **ipv6**] | **prefix** *ipaddress* | **range** *min-ipaddress max-ipaddress*}
6. **revocation-check** {[**crl**] [**none**] [**ocsp**]}
7. exit
8. **crypto pki server** *name*
9. **grant auto**
10. **cdp-url** *url-name*
11. **no shutdown**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ip http server** **Example:** `Router(config)# ip http server`	Configures the HTTP server.
Step 4	**crypto pki trustpoint** *name* **Example:** `Router(config)# crypto pki trustpoint CA`	(Optional) Declares the trustpoint that your certificate server should use and enters ca-trustpoint configuration mode. • If you plan to use X.509 IP extensions, use this command. To automatically generate a CS trustpoint, go to Step 8 .
Step 5	**ip-extension** [**multicast** | **unicast**] {**inherit** [**ipv4** | **ipv6**] | **prefix** *ipaddress* | **range** *min-ipaddress max-ipaddress*} **Example:** `Router(ca-trustpoint)# ip-extension prefix 2001:100::/32`	(Optional) Specifies that the IP extensions are included in a certificate request either for enrollment or generation of a certificate authority (CA) for the Cisco IOS CA.
Step 6	**revocation-check** {[**crl**] [**none**] [**ocsp**]} **Example:** `Router(ca-trustpoint)# revocation-check crl`	(Optional) Sets one or more methods for revocation checking.
Step 7	exit **Example:** `Router(ca-trustpoint)# exit`	Returns to global configuration mode.
Step 8	**crypto pki server** *name* **Example:** `Router(config)# crypto pki server CA`	Configures the PKI server and places the router in server configuration mode.
Step 9	**grant auto** **Example:** `Router(config-server)# grant auto`	(Optional) Grants all certificate requests automatically.

	Command or Action	Purpose
Step 10	**cdp-url** *url-name* **Example:** `Router(config-server)# cdp-url http://209.165.202.129/CA.crl`	(Optional) Sets the URL name if the host is using a Certificate Revocation List (CRL).
Step 11	**no shutdown** **Example:** `Router(config-server)# no shutdown`	Enables the certificate server.

Configuring a Host to Enable SeND

SeND is available in host mode. Before you can configure SeND parameters in host mode, first configure the host using the following commands. Once the host has been configured, SeND parameters can be configured on it.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto key generate rsa** [**general-keys** | **usage-keys** | **signature** | **encryption**] [**label** *key-label*] [**exportable**][**modulus** *modulus-size*] [**storage** *devicename* **:**] [**on** *devicename* **:**]
4. **ipv6 cga modifier rsakeypair** *key-label* **sec-level** {**0** | **1**}
5. **crypto pki trustpoint** *name*
6. **enrollment** [**mode**] [**retry period** *minutes*] [**retry count** *number*] **url** *url* [pem]
7. **revocation-check {[crl] [none] [ocsp]}**
8. exit
9. **crypto pki authenticate** *name*
10. **ipv6 nd secured sec-level minimum** *value*
11. **interface** *type number*
12. **ipv6 cga rsakeypair** *key-label*
13. **ipv6 address** *ipv6-address* / *prefix-length* **link-local cga**
14. **ipv6 nd secured trustanchor** *trustanchor-name*
15. **ipv6 nd secured timestamp** {**delta** *value* | **fuzz** *value*}
16. **exit**
17. **ipv6 nd secured full-secure**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Host> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Host# configure terminal`	Enters global configuration mode.
Step 3	**crypto key generate rsa** [**general-keys** | **usage-keys** | **signature** | **encryption**] [**label** *key-label*] [**exportable**][**modulus** *modulus-size*] [**storage** *devicename* **:**] [**on** *devicename* **:**] **Example:** `Host(config)# crypto key generate rsa label SEND modulus 1024`	Configures the RSA key.
Step 4	**ipv6 cga modifier rsakeypair** *key-label* **sec-level** {**0** | **1**} **Example:** `Host(config)# ipv6 cga modifier rsakeypair SEND sec-level 1`	Enables the RSA key to be used by SeND (generates the modifier).
Step 5	**crypto pki trustpoint** *name* **Example:** `Host(config)# crypto pki trustpoint SEND`	Specifies the node trustpoint and enters ca-trustpoint configuration mode.
Step 6	**enrollment** [**mode**] [**retry period** *minutes*] [**retry count** *number*] **url** *url* [pem] **Example:** `Host(ca-trustpoint)# enrollment url http://209.165.200.254`	Specifies the enrollment parameters of a CA.
Step 7	**revocation-check {[crl] [none] [ocsp]}** **Example:** `Host(ca-trustpoint)# revocation-check none`	Sets one or more methods of revocation.

	Command or Action	Purpose
Step 8	exit **Example:** `Host(ca-trustpoint)# exit`	Returns to global configuration mode.
Step 9	**crypto pki authenticate** *name* **Example:** `Host(config)# crypto pki authenticate SEND`	Authenticates the certification authority (by getting the certificate of the CA).
Step 10	**ipv6 nd secured sec-level minimum** *value* **Example:** `Host(config)# ipv6 nd secured sec-level minimum 1`	(Optional) Configures CGA. • You can provide additional parameters such as security level and key size. • In the example, the security level accepted by peers is configured.
Step 11	**interface** *type number* **Example:** `Host(config)# interface fastethernet 0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 12	**ipv6 cga rsakeypair** *key-label* **Example:** `Host(config-if)# ipv6 cga rsakeypair SEND`	(Optional) Configures CGA on interfaces.
Step 13	**ipv6 address** *ipv6-address* / *prefix-length* **link-local cga** **Example:** `Host(config-if)# ipv6 address FE80::260:3EFF:FE11:6770/23 link-local cga`	Configures an IPv6 link-local address for the interface and enables IPv6 processing on the interface.
Step 14	**ipv6 nd secured trustanchor** *trustanchor-name* **Example:** `Host(config-if)# ipv6 nd secured trustanchor SEND`	(Optional) Configures trusted anchors to be preferred for certificate validation.

	Command or Action	Purpose
Step 15	**ipv6 nd secured timestamp** {**delta** *value* | **fuzz** *value*} **Example:** `Host(config-if)# ipv6 nd secured timestamp delta 300`	(Optional) Configures the timing parameters.
Step 16	**exit** **Example:** `Host(config-if)# exit`	Returns to global configuration mode.
Step 17	**ipv6 nd secured full-secure** **Example:** `Host(config)# ipv6 nd secured full-secure`	(Optional) Configures general SeND parameters. • In the example, secure mode is configured on SeND.

Configuring a Router to Enable SeND

SeND is available in the router mode. Perform this task before you can configure SeND parameters in router mode. Once the router has been configured, the SeND parameters can be configured on it.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto key generate rsa** [**general-keys** | **usage-keys** | **signature** | **encryption**] [**label** *key-label*] [**exportable**][**modulus** *modulus-size*] [**storage** *devicename:*] [**on** *devicename:*]
4. **ipv6 cga modifier rsakeypair** *key-label* **sec-level** {**0** | **1**}
5. **crypto pki trustpoint** *name*
6. **subject-name** [**attr** *tag*][**eq** | **ne** | **co** | **nc**] *string*
7. **rsakeypair** *key-label*
8. **revocation-check** {[**crl**][**none**][**ocsp**]}
9. exit
10. **crypto pki authenticate** *name*
11. **crypto pki enroll** *name*
12. **ipv6 nd secured sec-level minimum** *value*
13. **interface** *type number*
14. **ipv6 cga rsakeypair** *key-label*
15. **ipv6 address** *ipv6-address* / *prefix-length* **link-local cga**
16. **ipv6 nd secured trustanchor** *trustpoint-name*
17. **ipv6 nd secured timestamp** {**delta** *value* | **fuzz** *value*}
18. **exit**
19. **ipv6 nd secured full-secure**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**crypto key generate rsa** [**general-keys** | **usage-keys** | **signature** | **encryption**] [**label** *key-label*] [**exportable**][**modulus** *modulus-size*] [**storage** *devicename:*] [**on** *devicename:*] **Example:** `Router(config)# crypto key generate rsa label SEND modulus 1024`	Configures the RSA key.
Step 4	**ipv6 cga modifier rsakeypair** *key-label* **sec-level** {**0** | **1**} **Example:** `Router(config)# ipv6 cga modifier rsakeypair SEND sec-level 1`	Enables the RSA key to be used by SeND (generates the modifier).
Step 5	**crypto pki trustpoint** *name* **Example:** `Router(config)# crypto pki trustpoint SEND`	Configures PKI for a single or multiple-tier CA, specifies the router trustpoint, and places the router in ca-trustpoint configuration mode.
Step 6	**subject-name** [**attr** *tag*][**eq** | **ne** | **co** | **nc**] *string* **Example:** `Router(ca-trustpoint)# subject-name C=FR, ST=PACA, L=Example, O=Cisco, OU=NSSTG, CN=router`	Creates a rule entry.
Step 7	**rsakeypair** *key-label* **Example:** `Router(ca-trustpoint)# rsakeypair SEND`	Binds the RSA key pair for SeND.
Step 8	**revocation-check** {[**crl**][**none**][**ocsp**]} **Example:** `Router(ca-trustpoint)# revocation-check none`	Sets one or more methods of revocation.
Step 9	exit **Example:** `host(ca-trustpoint)# exit`	Returns to global configuration mode.

	Command or Action	Purpose
Step 10	**crypto pki authenticate** *name* **Example:** `host(config)# crypto pki authenticate SEND`	Authenticates the certification authority (by getting the certificate of the CA).
Step 11	**crypto pki enroll** *name* **Example:** `Router(config)# crypto pki enroll SEND`	Obtains the certificates for the router from the CA.
Step 12	**ipv6 nd secured sec-level minimum** *value* **Example:** `Router(config)# ipv6 nd secured sec-level minimum 1`	(Optional) Configures CGA and provides additional parameters such as security level and key size. • In the example, the minimum security level that SeND accepts from its peers is configured.
Step 13	**interface** *type number* **Example:** `Router(config)# interface fastethernet 0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 14	**ipv6 cga rsakeypair** *key-label* **Example:** `Router(config-if)# ipv6 cga rsakeypair SEND`	(Optional) Configures CGA on interfaces. • In the example, CGA is generated.
Step 15	**ipv6 address** *ipv6-address* / *prefix-length* **link-local cga** **Example:** `Router(config-if)# ipv6 address fe80::link-local cga`	Configures an IPv6 link-local address for the interface and enables IPv6 processing on the interface.
Step 16	**ipv6 nd secured trustanchor** *trustpoint-name* **Example:** `Router(config-if)# ipv6 nd secured trustanchor SEND`	(Optional) Configures trusted anchors to be preferred for certificate validation.

	Command or Action	Purpose
Step 17	**ipv6 nd secured timestamp** {**delta** *value* | **fuzz** *value*} **Example:** `Router(config-if)# ipv6 nd secured timestamp delta 300`	(Optional) Configures the timing parameters.
Step 18	**exit** **Example:** `Router(config-if)# exit`	Returns to global configuration mode.
Step 19	**ipv6 nd secured full-secure** **Example:** `Router(config)# ipv6 nd secured full-secure`	(Optional) Configures general SeND parameters, such as secure mode and authorization method. • In the example, SeND security mode is enabled.

Implementing IPv6 SeND

Creating the RSA Key Pair and CGA Modifier for the Key Pair

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto key generate rsa** [**general-keys** | **usage-keys** | **signature** | **encryption**] [**label** *key-label*] [**exportable**] [**modulus** *modulus-size*] [**storage** *devicename* **:**] [**on** *devicename* **:**
4. **ipv6 cga modifier rsakeypair** *key-label* **sec-level** {**0** | **1**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto key generate rsa** [**general-keys** | **usage-keys** | **signature** | **encryption**] [**label** *key-label*] [**exportable**] [**modulus** *modulus-size*] [**storage** *devicename* **:**] [**on** *devicename* **:** **Example:** `Router(config)# crypto key generate rsa label SeND`	Generates RSA key pairs.
Step 4	**ipv6 cga modifier rsakeypair** *key-label* **sec-level** {**0** | **1**} **Example:** `Router(config)# ipv6 cga modifier rsakeypair SeND sec-level 1`	Generates the CGA modifier for a specified RSA key, which enables the key to be used by SeND.

Configuring Certificate Enrollment for a PKI

Certificate enrollment, which is the process of obtaining a certificate from a CA, occurs between the end host that requests the certificate and the CA. Each peer that participates in the PKI must enroll with a CA. In IPv6, you can autoenroll or manually enroll the device certificate.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki trustpoint** *name*
4. **subject-name** *x.500-name*]
5. **enrollment** [**mode**] [**retry period** *minutes*] [**retry count** *number*] **url** *url* [pem]
6. **serial-number** [**none**]
7. **auto-enroll** [*percent*] [**regenerate**
8. **password** *string*
9. **rsakeypair** *key-label key-size encryption-key-size*]]
10. **fingerprint** *ca-fingerprint*
11. **ip-extension** [**multicast** | **unicast**] {**inherit** [**ipv4** | **ipv6**] | **prefix** *ipaddress* | **range** *min-ipaddress max-ipaddress*}
12. **exit**
13. **crypto pki authenticate** *name*
14. **exit**
15. **copy** [**/ erase**] [**/ verify** | **/ noverify**] *source-url destination-url*
16. show crypto pki certificates
17. show crypto pki trustpoints [status | label [status]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki trustpoint** *name* **Example:** `Router(config)# crypto pki trustpoint trustpoint1`	Declares the trustpoint that your router should use and enters ca-trustpoint configuration mode.

	Command or Action	Purpose
Step 4	**subject-name** *x.500-name*] **Example:** `Router(ca-trustpoint)# subject-name name1`	Specifies the subject name in the certificate request.
Step 5	**enrollment** [**mode**] [**retry period** *minutes*] [**retry count** *number*] **url** *url* [pem] **Example:** `Router(ca-trustpoint)# enrollment url http://name1.example.com`	Specifies the URL of the CA on which your router should send certificate requests.
Step 6	**serial-number** [**none**] **Example:** `Router(ca-trustpoint)# serial-number`	(Optional) Specifies the router serial number in the certificate request.
Step 7	**auto-enroll** [*percent*] [**regenerate** **Example:** `Router(ca-trustpoint)# auto-enroll`	(Optional) Enables autoenrollment, allowing you to automatically request a router certificate from the CA.
Step 8	**password** *string* **Example:** `Router(ca-trustpoint)# password password1`	(Optional) Specifies the revocation password for the certificate.
Step 9	**rsakeypair** *key-label key-size encryption-key-size*]] **Example:** `Router(ca-trustpoint)# rsakeypair SEND`	Specifies which key pair to associate with the certificate.
Step 10	**fingerprint** *ca-fingerprint* **Example:** `Router(ca-trustpoint)# fingerprint 12EF53FA 355CD23E 12EF53FA 355CD23E`	(Optional) Specifies a fingerprint that can be matched against the fingerprint of a CA certificate during authentication.

Command or Action	Purpose
Step 11 **ip-extension** [**multicast** | **unicast**] {**inherit** [**ipv4** | **ipv6**] | **prefix** *ipaddress* | **range** *min-ipaddress max-ipaddress*} **Example:** `Router(ca-trustpoint)# ip-extension unicast prefix 2001:100:1://48`	Add IP extensions (IPv6 prefixes or range) to verify prefix list the router is allowed to advertise.
Step 12 **exit** **Example:** `Router(ca-trustpoint)# exit`	Exits ca-trustpoint configuration mode and returns to global configuration mode.
Step 13 **crypto pki authenticate** *name* **Example:** `Router(config)# crypto pki authenticate name1`	Retrieves and authenticates the CA certificate. • This command is optional if the CA certificate is already loaded into the configuration.
Step 14 **exit** **Example:** `Router(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 15 **copy** [**/ erase**] [**/ verify** | **/ noverify**] *source-url destination-url* **Example:** `Router# copy system:running-config nvram:startup-config`	(Optional) Copies the running configuration to the NVRAM startup configuration.
Step 16 show crypto pki certificates **Example:** `Router# show crypto pki certificates`	(Optional) Displays information about your certificates, the certificates of the CA, and RA certificates.
Step 17 show crypto pki trustpoints [status | label [status]] **Example:** `Router# show crypto pki trustpoints name1`	(Optional) Displays the trustpoints configured in the router.

Configuring a Cryptographically Generated Address

Configuring General CGA Parameters

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 nd secured sec-level** [**minimum** *value*]
4. **ipv6 nd secured key-length** [[**minimum** | **maximum**] *v alue*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 nd secured sec-level** [**minimum** *value*] **Example:** `Router(config)# ipv6 nd secured sec-level minimum 1`	Configures the SeND security level.
Step 4	**ipv6 nd secured key-length** [[**minimum**	**maximum**] *v alue* **Example:** `Router(config)# ipv6 nd secured key-length minimum 512`	Configures SeND key-length options.

Configuring CGA Address Generation on an Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 cga rsakeypair** *key-label*
5. **ipv6 address** {*ipv6-address* / *prefix-length* [**cga**] | *prefix-name sub-bits*/*prefix-length*[**cga**]}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface Ethernet 0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**ipv6 cga rsakeypair** *key-label* **Example:** `Router(config-if)# ipv6 cga rsakeypair SEND`	Specifies which RSA key pair should be used on a specified interface.
Step 5	**ipv6 address** {*ipv6-address* / *prefix-length* [**cga**]	*prefix-name sub-bits*/*prefix-length*[**cga**]} **Example:** `Router(config-if)# ipv6 address 2001:0DB8:1:1::/64 cga`	Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface. • The **cga** keyword generates a CGA address. **Note** The CGA link-local addresses must be configured by using the **ipv6 address link-local**command.

Configuring SeND Parameters

• Configuring the SeND Trustpoint, page 351

Configuring the SeND Trustpoint

In router mode, the key pair used to generate the CGA addresses on an interface must be certified by the CA and the certificate sent on demand over the SeND protocol. One RSA key pair and associated certificate is enough for SeND to operate; however, users may use several keys, identified by different labels. SeND and CGA refer to a key directly by label or indirectly by trustpoint.

Multiple steps are required to bind SeND to a trustpoint. First, a key pair is generated. Then the device refers to it in a trustpoint, and next the SeND interface configuration points to the trustpoint. There are two reasons for the multiple steps:

- The same key pair can be used on several SeND interfaces
- The trustpoint contains additional information, such as the certificate, required for SeND to perform authorization delegation

A CA certificate must be uploaded for the referred trustpoint. The referred trustpoint is in reality a trusted anchor.

Several trustpoints can be configured, pointing to the same RSA keys, on a given interface. This function is useful if different hosts have different trusted anchors (that is, CAs that they trust). The router can the provide each host with the certificate signed by the CA they trust.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto key generate rsa** [**general-keys** | **usage-keys** | **signature** | **encryption**] [**label** *key-label*] [**exportable**] [**modulus** *modulus-size*] [**storage** *devicename* **:**] [**on** *devicename* **:**
4. **ipv6 cga modifier rsakeypair** *key-label* **sec-level** {**0** | **1**}
5. **crypto pki trustpoint** *name*
6. **subject-name** [*x.500-name*]
7. **rsakeypair** *key-label key-size encryption-key-size*]]
8. **enrollment terminal** [**pem**
9. **ip-extension** [**multicast** | **unicast**] {**inherit** [**ipv4** | **ipv6**] | **prefix** *ipaddress* | **range** *min-ipaddress max-ipaddress*}
10. **exit**
11. **crypto pki authenticate** *name*
12. **crypto pki enroll** *name*
13. **crypto pki import** *name* **certificate**
14. **interface** *type number*
15. **ipv6 nd secured trustpoint** *trustpoint-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto key generate rsa** [**general-keys** | **usage-keys** | **signature** | **encryption**] [**label** *key-label*] [**exportable**] [**modulus** *modulus-size*] [**storage** *devicename* **:**] [**on** *devicename* **:** **Example:** `Router(config)# crypto key generate rsa label SEND`	Generates RSA key pairs.
Step 4	**ipv6 cga modifier rsakeypair** *key-label* **sec-level** {**0** | **1**} **Example:** `Router(config)# ipv6 cga modifier rsakeypair SEND sec-level 1`	Generates the CGA modifier for a specified RSA key, which enables the key to be used by SeND.
Step 5	**crypto pki trustpoint** *name* **Example:** `Router(config)# crypto pki trustpoint trustpoint1`	Declares the trustpoint that your router should use and enters ca-trustpoint configuration mode.
Step 6	**subject-name** [*x.500-name*] **Example:** `Router(ca-trustpoint)# subject-name name1`	Specifies the subject name in the certificate request.
Step 7	**rsakeypair** *key-label key-size encryption-key-size*]] **Example:** `Router(ca-trustpoint)# rsakeypair SEND`	Specifies which key pair to associate with the certificate.

	Command or Action	Purpose
Step 8	**enrollment terminal** [**pem**] **Example:** `Router(ca-trustpoint)# enrollment terminal`	Specifies manual cut-and-paste certificate enrollment.
Step 9	**ip-extension** [**multicast** | **unicast**] {**inherit** [**ipv4** | **ipv6**] | **prefix** *ipaddress* | **range** *min-ipaddress max-ipaddress*} **Example:** `Router(ca-trustpoint)# ip-extension unicast prefix 2001:100:1://48`	Adds IP extensions to the router certificate request.
Step 10	**exit** **Example:** `Router(ca-trustpoint)# exit`	Exits ca-trustpoint configuration mode and returns to global configuration mode.
Step 11	**crypto pki authenticate** *name* **Example:** `Router(config)# crypto pki authenticate trustpoint1`	Authenticates the certification authority (by getting the certificate of the CA).
Step 12	**crypto pki enroll** *name* **Example:** `Router(config)# crypto pki enroll trustpoint1`	Obtains the certificates for your router from the CA.
Step 13	**crypto pki import** *name* **certificate** **Example:** `Router(config)# crypto pki import trustpoint1 certificate`	Imports a certificate manually via TFTP or as a cut-and-paste at the terminal.
Step 14	**interface** *type number* **Example:** `Router(config)# interface Ethernet 0/0`	Specifies an interface type and number, and places the router in interface configuration mode.

	Command or Action	Purpose
Step 15	**ipv6 nd secured trustpoint** *trustpoint-name* **Example:** `Router(config-if)# ipv6 nd secured trustpoint trustpoint1`	Enables SeND on an interface and specifies which trustpoint should be used.

Configuring SeND Trust Anchors on the Interface

This task can be performed only in host mode. The host must be configured with one or more trust anchors. As soon as SeND is bound to a trustpoint on an interface (see Configuring the SeND Trustpoint, page 351), this trustpoint is also a trust anchor.

A trust anchor configuration consists of the following items:

- A public key signature algorithm and associated public key, which may include parameters
- A name
- An optional public key identifier
- An optional list of address ranges for which the trust anchor is authorized

Because PKI has already been configured, the trust anchor configuration is accomplished by binding SeND to one or several PKI trustpoints. PKI is used to upload the corresponding certificates, which contain the required parameters (such as name and key).

Perform this optional task to configure a trusted anchor on the interface. It allows you to select trust anchors listed in the CPS when requesting for a certificate. If you opt not to configure trust anchors, all the PKI trustpoints configured on the host will be considered.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki trustpoint** *name*
4. **enrollment terminal** [**pem**
5. **exit**
6. **crypto pki authenticate** *name*
7. **interface** *type number*
8. **ipv6 nd secured trustanchor** *trustanchor-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki trustpoint** *name* **Example:** `Router(config)# crypto pki trustpoint anchor1`	Declares the trustpoint that your router should use and enters ca-trustpoint configuration mode.
Step 4	**enrollment terminal** [**pem** **Example:** `Router(ca-trustpoint)# enrollment terminal`	Specifies manual cut-and-paste certificate enrollment.
Step 5	**exit** **Example:** `Router(ca-trustpoint)# exit`	Returns to global configuration.
Step 6	**crypto pki authenticate** *name* **Example:** `Router(config)# crypto pki authenticate anchor1`	Authenticates the certification authority (by getting the certificate of the CA).
Step 7	**interface** *type number* **Example:** `Router(config)# interface Ethernet 0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 8	**ipv6 nd secured trustanchor** *trustanchor-name* **Example:** `Router(config-if)# ipv6 nd secured trustanchor anchor1`	Specifies a trusted anchor on an interface and binds SeND to a trustpoint.

Configuring Secured and Nonsecured Neighbor Discovery Message Coexistence Mode

During the transition to SeND secured interfaces, network operators may want to run a particular interface with a mixture of nodes accepting secured and unsecured neighbor discovery messages. Perform this task to configure the coexistence mode for secure and nonsecure ND messages on the same interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 nd secured trustpoint** *trustpoint-name*
5. **no ipv6 nd secured full-secure**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface Ethernet 0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**ipv6 nd secured trustpoint** *trustpoint-name* **Example:** `Router(config-if)# ipv6 nd secured trustpoint trustpoint1`	Enables SeND on an interface and specifies which trustpoint should be used.
Step 5	**no ipv6 nd secured full-secure** **Example:** `Router(config-if)# no ipv6 nd secured full-secure`	Provides the coexistence mode for secure and nonsecure ND messages on the same interface.

Configuring SeND Parameters Globally

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 nd secured key-length** [[**minimum**| **maximum**] *value*]
4. **ipv6 nd secured sec-level minimum** *value*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 nd secured key-length** [[**minimum**| **maximum**] *value*] **Example:** `Router(config)# ipv6 nd secured key-length minimum 512`	Configures the SeND key-length options.
Step 4	**ipv6 nd secured sec-level minimum** *value* **Example:** `Router(config)# ipv6 nd secured sec-level minimum 2`	Configures the minimum security level value that can be accepted from peers.

Configuring the SeND Time Stamp

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 nd secured timestamp** {**delta** *value* | **fuzz** *value*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface Ethernet 0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**ipv6 nd secured timestamp** {**delta** *value* | **fuzz** *value* **Example:** `Router(config-if)# ipv6 nd secured timestamp delta 600`	Configures the SeND time stamp.

Configuring IPv6 PACL

Creating an IPv6 Access List

The first task in configuring IPv6 PACL is to create an IPv6 access list. This task is described in detail in Implementing Traffic Filters and Firewalls for IPv6 Security.

Configuring PACL Mode and Applying IPv6 PACL on an Interface

Before you configure the IPv6 PACL feature, you must configure an IPv6 access list. Once you have configured the IPv6 access list you want to use, you must configure the PACL mode on the specified IPv6 Layer 2 interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **access-group mode** {**prefer** {**port** | **vlan**} | **merge**}
5. **ipv6 traffic-filter** *access-list-name* {**in** | **out**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface fastethernet 0/0`	Specifies an interface type and number, and places the device in interface configuration mode.
Step 4	**access-group mode** {**prefer** {**port**	**vlan**}	**merge**} **Example:** `Device(config-if)# access-group mode prefer port`	Configures the mode for the specified Layer 2 interface. • The **no** form of this command sets the mode to the default value, which is **merge**. • The **prefer vlan** keyword combination is not supported in IPv6.
Step 5	**ipv6 traffic-filter** *access-list-name* {**in**	**out**} **Example:** `Device(config-if)# ipv6 traffic-filter list1 in`	Filters incoming IPv6 traffic on an interface. **Note** The **out** keyword and therefore filtering of outgoing traffic is not supported in IPv6 PACL configuration.

Configuration Examples for Implementing First Hop Security in IPv6

Example: IPv6 ND Inspection and RA Guard Configuration

This example provides information about an interface on which both the neighbor discovery (ND) inspection and router advertisement (RA) guard features are configured:

```
Device# show ipv6 snooping capture-policy interface ethernet 0/0

Hardware policy registered on Ethernet 0/0
Protocol     Protocol value   Message   Value     Action    Feature
ICMP         58               RS        85        punt      RA Guard
                                                  punt      ND Inspection
ICMP         58               RA        86        drop      RA guard
                                                  punt      ND Inspection
ICMP         58               NS        87        punt      ND Inspection
ICM          58               NA        88        punt      ND Inspection
ICMP         58               REDIR     89        drop      RA Guard
                                                  punt      ND Inspection
```

Example: IPv6 RA Guard Configuration

```
Device(config)# interface fastethernet 3/13

Device(config-if)# ipv6 nd raguard attach-policy

Device# show running-config interface fastethernet 3/13

Building configuration...
Current configuration : 129 bytes
!
interface FastEthernet3/13
 switchport
 switchport access vlan 222
 switchport mode access
 access-group mode prefer port
 ipv6 nd raguard
end
```

Example: Configuring PACL Mode and Applying IPv6 PACL on an Interface

```
Device(config)# interface gigabitethernet 3/24
Device(config-if)# access-group mode prefer port
Device(config-if)# ipv6 traffic-filter list1 in
```

Example SeND Configuration Examples

Example Configuring Certificate Servers

The following example shows how to configure certificate servers:

```
crypto pki server CA
 issuer-name C=FR, ST=fr, L=example, O=cisco, OU=nsstg, CN=CA0  lifetime ca-certificate
700 !
crypto pki trustpoint CA
 ip-extension prefix 2001::/16
 revocation-check crl
 rsakeypair CA
no shutdown
```

Note If you need to configure certificate servers without IP extensions, do not use the **ip-extension**command.

To display the certificate servers with IP extensions, use the **show crypto pki certificates verbose** command:

```
Router# show crypto pki certificates verbose
CA Certificate
  Status: Available
  Version: 3
  Certificate Serial Number (hex): 01
  Certificate Usage: Signature
  Issuer:
    c=FR
    st=fr
    l=example
    o=cisco
    ou=nsstg
    cn=CA0
  Subject:
    c=FR
    st=fr
    l=example
    o=cisco
    ou=nsstg
    cn=CA0
  Validity Date:
    start date: 09:50:52 GMT Feb 5 2009
    end   date: 09:50:52 GMT Jan 6 2011
  Subject Key Info:
    Public Key Algorithm: rsaEncryption
    RSA Public Key: (1024 bit)
  Signature Algorithm: MD5 with RSA Encryption
  Fingerprint MD5: 87DB764F 29367A65 D05CEE3D C12E0AC3
  Fingerprint SHA1: 04A06602 86AA72E9 43F2DB33 4A7D40A2 E2ED3325
  X509v3 extensions:
    X509v3 Key Usage: 86000000
      Digital Signature
      Key Cert Sign
      CRL Signature
    X509v3 Subject Key ID: 75B477C6 B2CA7BBE C7866657 57C84A32 90CEFB5A
    X509v3 Basic Constraints:
        CA: TRUE
    X509v3 Authority Key ID: 75B477C6 B2CA7BBE C7866657 57C84A32 90CEFB5A
    Authority Info Access:
```

```
    X509v3 IP Extension:
        IPv6:
          2001::/16
  Associated Trustpoints: CA
```

Example Configuring a Host to Enable SeND

The following example shows how to configure a host to enable SeND:

```
enable
configure terminal
 crypto key generate rsa label SEND modulus 1024
The name for the keys will be: SEND
% The key modulus size is 1024 bits
% Generating 1024 bit RSA keys, keys will be non-exportable...[OK]
 ipv6 cga modifier rsakeypair SEND sec-level 1
 crypto pki trustpoint SEND
  enrollment url http://209.165.200.254
  revocation-check none
  exit
 crypto pki authenticate SEND
Certificate has the following attributes:
    Fingerprint MD5: FC99580D 0A280EB4 2EB9E72B 941E9BDA
    Fingerprint SHA1: 22B10EF0 9A519177 145EA4F6 73667837 3A154C53
% Do you accept this certificate? [yes/no]: yes
Trustpoint CA certificate accepted.
 ipv6 nd secured sec-level minimum 1
 interface fastethernet 0/0
  ipv6 cga rsakeypair SEND
  ipv6 address FE80::260:3EFF:FE11:6770 link-local cga
  ipv6 nd secured trustanchor SEND
  ipv6 nd secured timestamp delta 300
  exit
 ipv6 nd secured full-secure
```

To verify the configuration use the **show running-config** command:

```
host# show running-config
Building configuration...
[snip]
crypto pki trustpoint SEND
 enrollment url http://209.165.200.225
 revocation-check none
!
interface Ethernet1/0
 ip address 209.165.202.129 255.255.255.0
 duplex half
 ipv6 cga rsakeypair SEND
 ipv6 address 2001:100::/64 cga
```

Example Configuring a Router to Enable SeND

The following example shows how to configure the router to enable SeND:

```
enable
configure terminal
 crypto key generate rsa label SEND modulus 1024
 ipv6 cga modifier rsakeypair SEND sec-level 1
 crypto pki trustpoint SEND
  subject-name C=FR, ST=PACA, L=Example, O=Cisco, OU=NSSTG, CN=router
  rsakeypair SEND
  revocation-check none
  exit
 crypto pki authenticate SEND
Certificate has the following attributes:
    Fingerprint MD5: FC99580D 0A280EB4 2EB9E72B 941E9BDA
    Fingerprint SHA1: 22B10EF0 9A519177 145EA4F6 73667837 3A154C53
% Do you accept this certificate? [yes/no]: yes
```

```
Trustpoint CA certificate accepted.
 crypto pki enroll SEND
% Start certificate enrollment ..
% Create a challenge password. You will need to verbally provide this
   password to the CA Administrator in order to revoke your certificate.
   For security reasons your password will not be saved in the configuration.
   Please make a note of it.
Password:
Re-enter password:
% The subject name in the certificate will include: C=FR, ST=fr, L=example, O=cisco,
OU=nsstg, CN=route r % The subject name in the certificate will include: Router % Include
the router serial number in the subject name? [yes/no]: no % Include an IP address in the
subject name? [no]:
Request certificate from CA? [yes/no]: yes % Certificate request sent to Certificate
Authority % The 'show crypto pki certificate SEND verbose' commandwill show the
fingerprint.
*Feb  5 09:40:37.171: CRYPTO_PKI:  Certificate Request Fingerprint MD5:
A6892F9F 23561949 4CE96BB8 CBC85 E64
*Feb  5 09:40:37.175: CRYPTO_PKI:  Certificate Request Fingerprint SHA1:
30832A66 E6EB37DF E578911D 383F 96A0 B30152E7
*Feb  5 09:40:39.843: %PKI-6-CERTRET: Certificate received from Certificate Authority
 interface fastethernet 0/0
  ipv6 nd secured sec-level minimum 1
  ipv6 cga rsakeypair SEND
  ipv6 address fe80::link-local cga
  ipv6 nd secured trustanchor SEND
  ipv6 nd secured timestamp delta 300
  exit
 ipv6 nd secured full-secure
```

To verify that the certificates are generated, use the **show crypto pki certificates** command:

```
Router# show crypto pki certificates
Certificate
  Status: Available
  Certificate Serial Number: 0x15
  Certificate Usage: General Purpose
  Issuer:
    cn=CA
  Subject:
    Name: Router
    hostname=Router
    c=FR
    st=fr
    l=example
    o=cisco
    ou=nsstg
    cn=router
  Validity Date:
    start date: 09:40:38 UTC Feb 5 2009
    end   date: 09:40:38 UTC Feb 5 2010
  Associated Trustpoints: SEND
CA Certificate
  Status: Available
  Certificate Serial Number: 0x1
  Certificate Usage: Signature
  Issuer:
    cn=CA
  Subject:
    cn=CA
  Validity Date:
    start date: 10:54:53 UTC Jun 20 2008
    end   date: 10:54:53 UTC Jun 20 2011
  Associated Trustpoints: SEND
```

To verify the configuration, use the **show running-config** command:

```
Router# show running-config
Building configuration...
[snip]
crypto pki trustpoint SEND
 enrollment url http://209.165.201.1
 subject-name C=FR, ST=fr, L=example, O=cisco, OU=nsstg, CN=router  revocation-check
```

```
none  rsakeypair SEND !
interface Ethernet1/0
 ip address 209.165.200.225 255.255.255.0
 duplex half
 ipv6 cga rsakeypair SEND
 ipv6 address FE80:: link-local cga
 ipv6 address 2001:100::/64 cga
```

Example Configuring a SeND Trustpoint in Router Mode

The following example shows how to configure a SeND trustpoint in router mode:

```
enable
 configure terminal
 crypto key generate rsa label SEND
  Choose the size of the key modulus in the range of 360 to 2048 for your
 General Purpose Keys. Choosing a key modulus greater than 512 may take
  a few minutes.
How many bits in the modulus [512]: 778
% Generating 778 bit RSA keys, keys will be non-exportable...[OK]
 ipv6 cga modifier rsakeypair SEND sec-level 1
 crypto pki trustpoint trstpt1
  subject-name C=FR, ST=fr, L=example, O=cisco, OU=nsstg, CN=sa14-72b
  rsakeypair SEND
  enrollment terminal
  ip-extension unicast prefix 2001:100:1://48
  exit
 crypto pki authenticate trstpt1
 crypto pki enroll trstpt1
 crypto pki import trstpt1 certificate
 interface Ethernet 0/0
  ipv6 nd secured trustpoint trstpt1
```

Example Configuring SeND Trust Anchors in the Host Mode

The following example shows how to configure SeND trust anchors on an interface in the host mode:

```
enable
configure terminal
! Configure the location of the CS we trust !
 crypto pki trustpoint B1
  enrollment terminal
  crypto pki authenticate anchor1
  exit
! Only Query a certificate signed by the CS at B2 on this interface !
 interface Ethernet0/0
  ip address 204.209.1.54 255.255.255.0
  ipv6 cga rsakeypair SEND
  ipv6 address 2001:100::/64 cga
  ipv6 nd secured trustanchor anchor1
```

Example Configuring CGA Address Generation on an Interface

The following example shows how to configure CGA address generation on an interface:

```
enable
 configure terminal
 interface fastEthernet 0/0
  ipv6 cga rsakeypair SEND
  ipv6 address 2001:100::/64 cga
  exit
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 Neighbor Discovery	Implementing IPv6 Addressing and Basic Connectivity
ICMP in IPv6	Implementing IPv6 Addressing and Basic Connectivity
IPv6--IPv6 stateless autoconfiguration	Implementing IPv6 Addressing and Basic Connectivity
IPv6 access lists	Implementing Traffic Filters and Firewalls for IPv6 Security
IPv6 DHCP	Implementing DHCP for IPv6
Configuring certificate enrollment for a PKI	"Configuring Certificate Enrollment for a PKI" module in the *Cisco IOS Security Configuration Guide*
IPv6 commands	*Cisco IOS IPv6 Command Reference*
All Cisco IOS commands	Cisco IOS Master Command List, All Releases

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	--

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 3756	*IPv6 Neighbor Discovery (ND) Trust Models and Threats*
RFC 3779	X.509 Extensions for IP Addresses and AS Identifiers
RFC 3971	*Secure Neighbor Discovery (SeND)*
RFC 3972	*Cryptographically Generated Addresses (CGA)*
RFC 6105	*IPv6 Router Advertisement Guard*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing First Hop Security in IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 17 Feature Information for Implementing First Hop Security in IPv6

Feature Name	Releases	Feature Information
IPv6 Device Tracking	12.2(50)SY	This feature allows IPv6 host liveness to be tracked so the neighbor binding table can be immediately updated when an IPv6 host disappears. The following commands were introduced or modified: **ipv6 neighbor binding**, **ipv6 neighbor binding down-lifetime**, **ipv6 neighbor binding logging**, **ipv6 neighbor binding max-entries**, **ipv6 neighbor binding stale-lifetime**, **ipv6 neighbor binding vlan**, **ipv6 neighbor tracking**, **show ipv6 neighbor binding**.
IPv6 ND Inspection	12.2(50)SY	The IPv6 ND Inspection feature learns and secures bindings for stateless autoconfiguration addresses in layer 2 neighbor tables. The following commands were introduced: clear ipv6 snooping counters, debug ipv6 snooping, **device-role**, **drop-unsecure**, **ipv6 nd inspection**, **ipv6 nd inspection policy**, **sec-level minimum**, **show ipv6 snooping capture-policy**, s**how ipv6 snooping counters**,s**how ipv6 snooping features**, **show ipv6 snooping policies**, **tracking trusted-port**.
IPv6 PACL	12.2(54)SG 12.2(33)SXI4 12.2(50)SY	The IPv6 PACL permits or denies the movement of traffic between Layer 3 (L3) subnets and VLANs, or within a VLAN. The following commands were introduced or modified: access-group mode, ipv6 traffic-filter.

Feature Name	Releases	Feature Information
IPv6 RA Guard	12.2(54)SG 12.2(33)SXI4 12.2(50)SY	IPv6 RA guard provides support for allowing the network administrator to block or reject unwanted or rogue RA guard messages that arrive at the network switch platform.

Feature Name	Releases	Feature Information
Secure Neighbor Discovery for Cisco IOS Software	12.4(24)T	The Secure Neighbor Discovery (SeND) protocol is designed to counter the threats of the ND protocol. SeND defines a set of neighbor discovery options and two neighbor discovery messages. SeND also defines a new autoconfiguration mechanism to establish address ownership. The following commands were introduced or modified: **auto-enroll**, **crypto key generate rsa**, **crypto pki authenticate**, **crypto pki enroll**, **crypto pki import**, **enrollment terminal (ca-trustpoint)**, **enrollment url (ca-trustpoint)**, **fingerprint**, **ip-extension**, **ip http server**, **ipv6 address**, **ipv6 address link-local**, **ipv6 cga modifier rsakeypair**, **ipv6 cga modifier rsakeypair (interface)**, **ipv6 nd secured certificate-db**, **ipv6 nd secured full-secure**, **ipv6 nd secured full-secure (interface)**, **ipv6 nd secured key-length**, **ipv6 nd secured sec-level**, **ipv6 nd secured timestamp**, **ipv6 nd secured timestamp-db**, **ipv6 nd secured trustanchor**, **ipv6 nd secured trustpoint**, **password (ca-trustpoint)**, **revocation-check**, **rsakeypair**, **serial-number (ca-trustpoint)**, **show ipv6 cga address-db**, **show ipv6 cga modifier-db**, **show ipv6 nd secured certificates**, **show ipv6 nd secured counters interface**, **show ipv6 nd secured nonce-db**, **show ipv6 nd secured timestamp-db**, **subject-name**.

Glossary

- **ACE** --access control entry
- **ACL** --access control list

- **CA** --certification authority.
- **CGA** --cryptographically generated address.
- **CPA** --certificate path answer.
- **CPR** --certificate path response.
- **CPS** --certification path solicitation. The solicitation message used in the addressing process.
- **CRL** --certificate revocation list.
- **CS** --certification server.
- **CSR** --certificate signing request.
- **DAD** --duplicate address detection. A mechanism that ensures two IPv6 nodes on the same link are not using the same address.
- **DER** --distinguished encoding rules. An encoding scheme for data values.
- **LLA** --link-layer address.
- **MAC** --media access control.
- **nonce** --An unpredictable random or pseudorandom number generated by a node and used once. In SeND, nonces are used to assure that a particular advertisement is linked to the solicitation that triggered it.
- **non-SeND node** --An IPv6 node that does not implement SeND but uses only the neighbor discovery protocol without security.
- **NUD** --neighbor unreachability detection. A mechanism used for tracking neighbor reachability.
- **PACL** --port-based access list.
- **PKI** --public key infrastructure.
- **RA** --router advertisement.
- **Router Authorization Certificate** --A public key certificate.
- **RD** --Router discovery allows the hosts to discover what routers exist on the link and what subnet prefixes are available. Router discovery is a part of the neighbor discovery protocol.
- **SeND node** --An IPv6 node that implements SeND.
- **trust anchor** --A trust anchor is an entity that the host trusts to authorize routers to act as routers. Hosts are configured with a set of trust anchors to protect router discovery.
- **ULA** --unique local addressing.

Implementing IPsec in IPv6 Security

Cisco IOS IPv6 security features for your Cisco networking devices can protect your network against degradation or failure and also against data loss or compromise resulting from intentional attacks and from unintended but damaging mistakes by well-meaning network users.

Cisco IOS IPsec functionality provides network data encryption at the IP packet level, offering a robust, standards-based security solution. IPsec provides data authentication and anti-replay services in addition to data confidentiality services.

IPsec is a mandatory component of IPv6 specification. OSPF for IPv6 provides IPsec authentication support and protection, and IPv6 IPsec tunnel mode and encapsulation is used to protect IPv6 unicast and multicast traffic. This document provides information about implementing IPsec in IPv6 security.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Implementing IPsec for IPv6 Security

IPsec for IPv6

IP Security, or IPsec, is a framework of open standards developed by the Internet Engineering Task Force (IETF) that provide security for transmission of sensitive information over unprotected networks such as the Internet. IPsec acts at the network layer, protecting and authenticating IP packets between participating

IPsec devices (peers), such as Cisco routers. IPsec provides the following optional network security services. In general, local security policy will dictate the use of one or more of these services:

- Data confidentiality--The IPsec sender can encrypt packets before sending them across a network.
- Data integrity--The IPsec receiver can authenticate packets sent by the IPsec sender to ensure that the data has not been altered during transmission.
- Data origin authentication--The IPsec receiver can authenticate the source of the IPsec packets sent. This service depends upon the data integrity service.
- Antireplay--The IPsec receiver can detect and reject replayed packets.

With IPsec, data can be sent across a public network without observation, modification, or spoofing. IPsec functionality is similar in both IPv6 and IPv4; however, site-to-site tunnel mode only is supported in IPv6.

In IPv6, IPsec is implemented using the AH authentication header and the ESP extension header. The authentication header provides integrity and authentication of the source. It also provides optional protection against replayed packets. The authentication header protects the integrity of most of the IP header fields and authenticates the source through a signature-based algorithm. The ESP header provides confidentiality, authentication of the source, connectionless integrity of the inner packet, antireplay, and limited traffic flow confidentiality.

The Internet Key Exchange (IKE) protocol is a key management protocol standard that is used in conjunction with IPsec. IPsec can be configured without IKE, but IKE enhances IPsec by providing additional features, flexibility, and ease of configuration for the IPsec standard.

IKE is a hybrid protocol that implements the Oakley key exchange and Skeme key exchange inside the Internet Security Association Key Management Protocol (ISAKMP) framework (ISAKMP, Oakley, and Skeme are security protocols implemented by IKE) (see the figure below). This functionality is similar to the security gateway model using IPv4 IPsec protection.

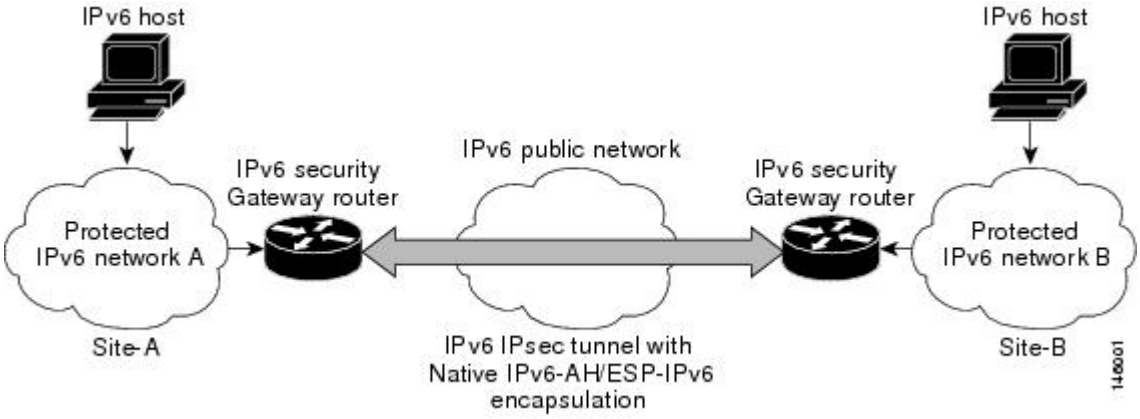
- IPv6 IPsec Site-to-Site Protection Using Virtual Tunnel Interface, page 372
- OSPFv3 Authentication Support with IPsec, page 373

IPv6 IPsec Site-to-Site Protection Using Virtual Tunnel Interface

The IPsec virtual tunnel interface (VTI) provides site-to-site IPv6 crypto protection of IPv6 traffic. Native IPv6 IPsec encapsulation is used to protect all types of IPv6 unicast and multicast traffic.

The IPsec VTI allows IPv6 routers to work as security gateways, establish IPsec tunnels between other security gateway routers, and provide crypto IPsec protection for traffic from internal networks when it is sent across the public IPv6 Internet (see the figure below). This functionality is similar to the security gateway model using IPv4 IPsec protection.

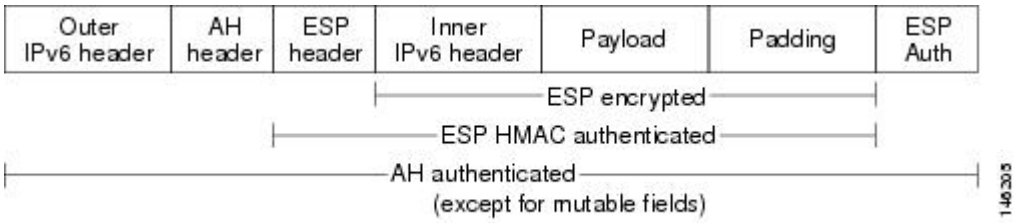
Figure 30 IPsec Tunnel Interface for IPv6



When the IPsec tunnel is configured, IKE and IPsec security associations (SAs) are negotiated and set up before the line protocol for the tunnel interface is changed to the UP state. The remote IKE peer is the same as the tunnel destination address; the local IKE peer will be the address picked from tunnel source interface which has the same IPv6 address scope as tunnel destination address.

The following figures shows the IPsec packet format.

Figure 31 IPv6 IPsec Packet Format



OSPFv3 Authentication Support with IPsec

In order to ensure that OSPFv3 packets are not altered and re-sent to the router, causing the router to behave in a way not desired by its system administrators, OSPFv3 packets must be authenticated. OSPFv3 uses the IPsec secure socket API to add authentication to OSPFv3 packets. This API supports IPv6.

OSPFv3 requires the use of IPsec to enable authentication. Crypto images are required to use authentication, because only crypto images include the IPsec API needed for use with OSPFv3.

In OSPFv3, authentication fields have been removed from OSPFv3 packet headers. When OSPFv3 runs on IPv6, OSPFv3 requires the IPv6 authentication header (AH) or IPv6 ESP header to ensure integrity, authentication, and confidentiality of routing exchanges. IPv6 AH and ESP extension headers can be used to provide authentication and confidentiality to OSPFv3.

To use the IPsec AH, you must enable the **ipv6 ospf authentication** command. To use the IPsec ESP header, you must enable the **ipv6 ospf encryption** command. The ESP header may be applied alone or in combination with the AH, and when ESP is used, both encryption and authentication are provided. Security services can be provided between a pair of communicating hosts, between a pair of communicating security gateways, or between a security gateway and a host.

To configure IPsec, you configure a security policy, which is a combination of the security policy index (SPI) and the key (the key is used to create and validate the hash value). IPsec for OSPFv3 can be configured on an interface or on an OSPFv3 area. For higher security, you should configure a different policy on each interface configured with IPsec. If you configure IPsec for an OSPFv3 area, the policy is applied to all of the interfaces in that area, except for the interfaces that have IPsec configured directly. Once IPsec is configured for OSPFv3, IPsec is invisible to you.

The secure socket API is used by applications to secure traffic. The API needs to allow the application to open, listen, and close secure sockets. The binding between the application and the secure socket layer also allows the secure socket layer to inform the application of changes to the socket, such as connection open and close events. The secure socket API is able to identify the socket; that is, it can identify the local and remote addresses, masks, ports, and protocol that carry the traffic requiring security.

Each interface has a secure socket state, which can be one of the following:

- NULL: Do not create a secure socket for the interface if authentication is configured for the area.

- DOWN: IPsec has been configured for the interface (or the area that contains the interface), but OSPFv3 either has not requested IPsec to create a secure socket for this interface, or there is an error condition.
- GOING UP: OSPFv3 has requested a secure socket from IPsec and is waiting for a CRYPTO_SS_SOCKET_UP message from IPsec.
- UP: OSPFv3 has received a CRYPTO_SS_SOCKET_UP message from IPsec.
- CLOSING: The secure socket for the interface has been closed. A new socket may be opened for the interface, in which case the current secure socket makes the transition to the DOWN state. Otherwise, the interface will become UNCONFIGURED.
- UNCONFIGURED: Authentication is not configured on the interface.

OSPFv3 will not send or accept packets while in the DOWN state.

How to Implement IPsec for IPv6 Security

Configuring a VTI for Site-to-Site IPv6 IPsec Protection

Creating an IKE Policy and a Preshared Key in IPv6

Because IKE negotiations must be protected, each IKE negotiation begins by agreement of both peers on a common (shared) IKE policy. This policy states which security parameters will be used to protect subsequent IKE negotiations and mandates how the peers are authenticated.

After the two peers agree upon a policy, the security parameters of the policy are identified by an SA established at each peer, and these SAs apply to all subsequent IKE traffic during the negotiation.

You can configure multiple, prioritized policies on each peer--each with a different combination of parameter values. However, at least one of these policies must contain exactly the same encryption, hash, authentication, and Diffie-Hellman parameter values as one of the policies on the remote peer. For each policy that you create, you assign a unique priority (1 through 10,000, with 1 being the highest priority).

Note If you are interoperating with a device that supports only one of the values for a parameter, your choice is limited to the value supported by the other device. Aside from this limitation, there is often a trade-off between security and performance, and many of these parameter values represent such a trade-off. You should evaluate the level of security risks for your network and your tolerance for these risks.

When the IKE negotiation begins, IKE searches for an IKE policy that is the same on both peers. The peer that initiates the negotiation will send all its policies to the remote peer, and the remote peer will try to find

a match. The remote peer looks for a match by comparing its own highest priority policy against the policies received from the other peer. The remote peer checks each of its policies in order of its priority (highest priority first) until a match is found.

A match is made when both policies from the two peers contain the same encryption, hash, authentication, and Diffie-Hellman parameter values, and when the remote peer's policy specifies a lifetime that is less than or equal to the lifetime in the policy being compared. (If the lifetimes are not identical, the shorter lifetime--from the remote peer's policy--will be used.)

If a match is found, IKE will complete negotiation, and IPsec security associations will be created. If no acceptable match is found, IKE refuses negotiation and IPsec will not be established.

Note Depending on which authentication method is specified in a policy, additional configuration might be required. If a peer's policy does not have the required companion configuration, the peer will not submit the policy when attempting to find a matching policy with the remote peer.

You should set the ISAKMP identity for each peer that uses preshared keys in an IKE policy.

When two peers use IKE to establish IPsec SAs, each peer sends its identity to the remote peer. Each peer sends either its hostname or its IPv6 address, depending on how you have set the ISAKMP identity of the router.

By default, a peer's ISAKMP identity is the IPv6 address of the peer. If appropriate, you could change the identity to be the peer's hostname instead. As a general rule, set the identities of all peers the same way--either all peers should use their IPv6 addresses or all peers should use their hostnames. If some peers use their hostnames and some peers use their IPv6 addresses to identify themselves to each other, IKE negotiations could fail if the identity of a remote peer is not recognized and a DNS lookup is unable to resolve the identity.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto isakmp policy** *priority*
4. **authentication** {**rsa-sig** | **rsa-encr** | **pre-share**}
5. **hash** {**sha** | **md5**}
6. **group** {**1** | **2** | **5**}
7. **encryption** {**des** | **3des** | **aes** | **aes 192** | **aes 256**}
8. **lifetime** *seconds*
9. **exit**
10. **crypto isakmp key** *enc-type-digit keystring {* **address** *peer-address [mask]* | **ipv6** {*ipv6-address*/*ipv6-prefix*} | **hostname** *hostname} [* **no-xauth**]
11. **crypto keyring** *keyring-name* [**vrf** *fvrf-name*]
12. **pre-shared-key** {**address** *address* [*mask*] | **hostname** *hostname* | **ipv6** {*ipv6-address* | *ipv6-prefix*}} **key** *key*
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto isakmp policy** *priority* **Example:** `Router(config)# crypto isakmp policy 15`	Defines an IKE policy, and enters ISAKMP policy configuration mode. Policy number 1 indicates the policy with the highest priority. The smaller the *priority* argument value, the higher the priority.
Step 4	**authentication** {**rsa-sig** | **rsa-encr** | **pre-share**} **Example:** `Router(config-isakmp-policy)# authentication pre-share`	Specifies the authentication method within an IKE policy. The **rsa-sig** and **rsa-encr** keywords are not supported in IPv6.
Step 5	**hash** {**sha** | **md5**} **Example:** `Router(config-isakmp-policy)# hash md5`	Specifies the hash algorithm within an IKE policy.
Step 6	**group** {**1** | **2** | **5**} **Example:** `Router(config-isakmp-policy)# group 2`	Specifies the Diffie-Hellman group identifier within an IKE policy.
Step 7	**encryption** {**des** | **3des** | **aes** | **aes 192** | **aes 256**} **Example:** `Router(config-isakmp-policy)# encryption 3des`	Specifies the encryption algorithm within an IKE policy.
Step 8	**lifetime** *seconds* **Example:** `Router(config-isakmp-policy)# lifetime 43200`	Specifies the lifetime of an IKE SA. Setting the IKE lifetime value is optional.

	Command or Action	Purpose
Step 9	**exit** **Example:** `Router(config-isakmp-policy)# exit`	Enter this command to exit ISAKMP policy configuration mode and enter global configuration mode.
Step 10	**crypto isakmp key** *enc-type-digit keystring {* **address** *peer-address [mask]* | **ipv6** {*ipv6-address/ipv6-prefix*} | **hostname** *hostname} [* **no-xauth**] **Example:** `Router(config)# crypto isakmp key 0 my-preshare-key-0 address ipv6 3ffe:1001::2/128`	Configures a preshared authentication key.
Step 11	**crypto keyring** *keyring-name* [**vrf** *fvrf-name*] **Example:** `Router(config)# crypto keyring keyring1`	Defines a crypto keyring to be used during IKE authentication.
Step 12	**pre-shared-key** {**address** *address* [*mask*] | **hostname** *hostname* | **ipv6** {*ipv6-address* | *ipv6-prefix*}} **key** *key* **Example:** `Router (config-keyring)# pre-shared-key ipv6 3FFE: 2002::A8BB:CCFF:FE01:2C02/128`	Defines a preshared key to be used for IKE authentication.
Step 13	**end** **Example:** `Router (config-keyring)# end`	Exits crypto keyring configuration mode and returns to privileged EXEC mode.

Configuring ISAKMP Aggressive Mode

You likely do not need to configure aggressive mode in a site-to-site scenario. The default mode is typically used.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto isakmp peer** {**address** {*ipv4-address* | **ipv6** *ipv6-address ipv6-prefix-length*} | **hostname** *fqdn-hostname*}
4. **set aggressive-mode client-endpoint** {*client-endpoint* | **ipv6** *ipv6-address*}
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto isakmp peer** {**address** {*ipv4-address* | **ipv6** *ipv6-address ipv6-prefix-length*} | **hostname** *fqdn-hostname*} **Example:** `Router(config)# crypto isakmp peer address ipv6 3FFE:2002::A8BB:CCFF:FE01:2C02/128`	Enables an IPsec peer for IKE querying for tunnel attributes.
Step 4	**set aggressive-mode client-endpoint** {*client-endpoint* | **ipv6** *ipv6-address*} **Example:** `Router(config-isakmp-peer)# set aggressive mode client-endpoint ipv6 3FFE:2002::A8BB:CCFF:FE01:2C02/128`	Defines the remote peer's IPv6 address, which will be used by aggressive mode negotiation. The remote peer's address is usually the client side's end-point address.
Step 5	**end** **Example:** `Router(config-isakmp-peer)# end`	Exits crypto ISAKMP peer configuration mode and returns to privileged EXEC mode.

Configuring an IPsec Transform Set and IPsec Profile

A transform set is a combination of security protocols and algorithms that is acceptable to the IPsec routers.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto ipsec transform-set** *transform-set-name transform1* [*transform2*] [*transform3*] [*transform4*]
4. **crypto ipsec profile** *name*
5. **set transform-set** *transform-set-name* [*transform-set-name2...transform-set-name6*]
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto ipsec transform-set** *transform-set-name transform1* [*transform2*] [*transform3*] [*transform4*] **Example:** `Router(config)# crypto ipsec transform-set myset0 ah-sha-hmac esp-3des`	Defines a transform set, and places the router in crypto transform configuration mode.
Step 4	**crypto ipsec profile** *name* **Example:** `Router(config)# crypto ipsec profile profile0`	Defines the IPsec parameters that are to be used for IPsec encryption between two IPsec routers.
Step 5	**set transform-set** *transform-set-name* [*transform-set-name2...transform-set-name6*] **Example:** `Router (config-crypto-transform)# set-transform-set myset0`	Specifies which transform sets can be used with the crypto map entry.
Step 6	**end** **Example:** `Router (config-crypto-transform)# end`	Exits crypto transform configuration mode and returns to privileged EXEC mode.

Defining an ISAKMP Profile in IPv6

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto isakmp profile** *profile-name* [**accounting** *aaalist*
4. **self-identity** {**address** | **address ipv6**] | **fqdn** | **user-fqdn** *user-fqdn*}
5. **match identity** {**group** *group-name* | **address** {*address* [*mask*] [*fvrf*] | **ipv6** *ipv6-address*} | **host** *host-name* | **host domain** *domain-name* | **user** *user-fqdn* | **user domain** *domain-name*}
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto isakmp profile** *profile-name* [**accounting** *aaalist* **Example:** `Router(config)# crypto isakmp profile profile1`	Defines an ISAKMP profile and audits IPsec user sessions.
Step 4	**self-identity** {**address**	**address ipv6**]	**fqdn**	**user-fqdn** *user-fqdn*} **Example:** `Router(config-isakmp-profile)# self-identity address ipv6`	Defines the identity that the local IKE uses to identify itself to the remote peer.
Step 5	**match identity** {**group** *group-name*	**address** {*address* [*mask*] [*fvrf*]	**ipv6** *ipv6-address*}	**host** *host-name*	**host domain** *domain-name*	**user** *user-fqdn*	**user domain** *domain-name*} **Example:** `Router(config-isakmp-profile)# match identity address ipv6 3FFE: 2002::A8BB:CCFF:FE01:2C02/128`	Matches an identity from a remote peer in an ISAKMP profile.

	Command or Action	Purpose
Step 6	**end** **Example:** `Router(config-isakmp-profile)# end`	Exits ISAKMP profile configuration mode and returns to privileged EXEC mode.

Configuring IPv6 IPsec VTI

Use the **ipv6 unicast-routing** command to enable IPv6 unicast routing.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **interface tunnel** *tunnel-number*
5. **ipv6 address** *ipv6-address/prefix*
6. **ipv6 enable**
7. **tunnel source** {*ip-address* | *ipv6-address* | *interface-type interface-number*}
8. **tunnel destination** {*host-name* | *ip-address* | *ipv6-address*}
9. **tunnel mode** {**aurp** | **cayman** | **dvmrp** | **eon** | **gre** | **gre multipoint** | **gre ipv6** | **ipip** [**decapsulate-any**] | **ipsec ipv4** | **iptalk** | **ipv6** | **ipsec ipv6** | **mpls** | **nos** | **rbscp**}
10. **tunnel protection ipsec profile** *name* [**shared**]
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ipv6 unicast-routing** **Example:** `Router(config)# ipv6 unicast-routing`	Enables IPv6 unicast routing. You only need to enable IPv6 unicast routing once, not matter how many interface tunnels you want to configure.
Step 4	**interface tunnel** *tunnel-number* **Example:** `Router(config)# interface tunnel 0`	Specifies a tunnel interface and number, and enters interface configuration mode.
Step 5	**ipv6 address** *ipv6-address/prefix* **Example:** `Router(config-if)# ipv6 address 3FFE:C000:0:7::/64 eui-64`	Provides an IPv6 address to this tunnel interface, so that IPv6 traffic can be routed to this tunnel.
Step 6	**ipv6 enable** **Example:** `Router(config-if)# ipv6 enable`	Enables IPv6 on this tunnel interface.
Step 7	**tunnel source** {*ip-address* | *ipv6-address* | *interface-type interface-number*} **Example:** `Router(config-if)# tunnel source ethernet0`	Sets the source address for a tunnel interface.
Step 8	**tunnel destination** {*host-name* | *ip-address* | *ipv6-address*} **Example:** `Router(config-if)# tunnel destination 2001:DB8:1111:2222::1`	Specifies the destination for a tunnel interface.
Step 9	**tunnel mode** {**aurp** | **cayman** | **dvmrp** | **eon** | **gre** | **gre multipoint** | **gre ipv6** | **ipip** [**decapsulate-any**] | **ipsec ipv4** | **iptalk** | **ipv6** | **ipsec ipv6** | **mpls** | **nos** | **rbscp**} **Example:** `Router(config-if)# tunnel mode ipsec ipv6`	Sets the encapsulation mode for the tunnel interface. For IPsec, only the **ipsec ipv6** keywords are supported.

	Command or Action	Purpose
Step 10	**tunnel protection ipsec profile** *name* [**shared**] **Example:** `Router(config-if)# tunnel protection ipsec profile profile1`	Associates a tunnel interface with an IPsec profile. IPv6 does not support the **shared** keyword.
Step 11	**end** **Example:** `Router(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Verifying IPsec Tunnel Mode Configuration

SUMMARY STEPS

1. **show adjacency** [**summary** [*interface-type interface-number*]] | [**prefix**] [**interface** *interface-number*] [**connectionid** *id*] [**link** {**ipv4**| **ipv6** | **mpls**}] [**detail**]
2. **show crypto engine** {**accelerator** | **brief** | **configuration** | **connections** [**active** | **dh** | **dropped-packet** | **show**] | **qos**}
3. **show crypto ipsec sa** [**ipv6**] [*interface-type interface-number*] [**detailed**]
4. **show crypto isakmp peer** [**config** | **detail**]
5. **show crypto isakmp policy**
6. **show crypto isakmp profile** [**tag** *profilename* | **vrf** *vrfname*]
7. **show crypto map** [**interface** *interface* | **tag** *map-name*]
8. **show crypto session** [**detail**] | [**local** *ip-address* [**port** *local-port*] | [**remote** *ip-address* [**port** *remote-port*]] | **detail**] | **fvfr** *vrf-name* | **ivrf** *vrf-name*]
9. **show crypto socket**
10. **show ipv6 access-list** [*access-list-name*]
11. **show ipv6 cef** [*ipv6-prefix / prefix-length*] | [*interface-type interface-number*] [**longer-prefixes** | **similar-prefixes** | **detail** | **internal** | **platform** | **epoch** | **source**]]
12. **show interface** *type number* **stats**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show adjacency** [**summary** [*interface-type interface-number*]] | [**prefix**] [**interface** *interface-number*] [**connectionid** *id*] [**link** {**ipv4**| **ipv6** | **mpls**}] [**detail**] **Example:** `Router# show adjacency detail`	Displays information about the Cisco Express Forwarding adjacency table or the hardware Layer 3-switching adjacency table.
Step 2	**show crypto engine** {**accelerator** | **brief** | **configuration** | **connections** [**active** | **dh** | **dropped-packet** | **show**] | **qos**} **Example:** `Router# show crypto engine connection active`	Displays a summary of the configuration information for the crypto engines.
Step 3	**show crypto ipsec sa** [**ipv6**] [*interface-type interface-number*] [**detailed**] **Example:** `Router# show crypto ipsec sa ipv6`	Displays the settings used by current SAs in IPv6.
Step 4	**show crypto isakmp peer** [**config** | **detail**] **Example:** `Router# show crypto isakmp peer detail`	Displays peer descriptions.
Step 5	**show crypto isakmp policy** **Example:** `Router# show crypto isakmp policy`	Displays the parameters for each IKE policy.
Step 6	**show crypto isakmp profile** [**tag** *profilename* | **vrf** *vrfname*] **Example:** `Router# show crypto isakmp profile`	Lists all the ISAKMP profiles that are defined on a router.

	Command or Action	Purpose
Step 7	**show crypto map** [**interface** *interface* | **tag** *map-name*] **Example:** `Router# show crypto map`	Displays the crypto map configuration. The crypto maps shown in this command output are dynamically generated. The user does not have to configure crypto maps.
Step 8	**show crypto session** [**detail**] | [**local** *ip-address* [**port** *local-port*] | [**remote** *ip-address* [**port** *remote-port*]] | **detail**] | **fvfr** *vrf-name* | **ivrf** *vrf-name*] **Example:** `Router# show crypto session`	Displays status information for active crypto sessions. IPv6 does not support the **fvfr** or **ivrf** keywords or the *vrf-name* argument.
Step 9	**show crypto socket** **Example:** `Router# show crypto socket`	Lists crypto sockets.
Step 10	**show ipv6 access-list** [*access-list-name*] **Example:** `Router# show ipv6 access-list`	Displays the contents of all current IPv6 access lists.
Step 11	**show ipv6 cef** [*ipv6-prefix / prefix-length*] | [*interface-type interface-number*] [**longer-prefixes** | **similar-prefixes** | **detail** | **internal** | **platform** | **epoch** | **source**]] **Example:** `Router# show ipv6 cef`	Displays entries in the IPv6 Forwarding Information Base (FIB).
Step 12	**show interface** *type number* **stats** **Example:** `Router# show interface fddi 3/0/0 stats`	Displays numbers of packets that were process switched, fast switched, and distributed switched.

Troubleshooting IPsec for IPv6 Configuration and Operation

SUMMARY STEPS

1. **enable**
2. **debug crypto ipsec**
3. **debug crypto engine packet** [**detail**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**debug crypto ipsec** **Example:** `Router# debug crypto ipsec`	Displays IPsec network events.
Step 3	**debug crypto engine packet** [**detail**] **Example:** `Router# debug crypto engine packet`	Displays the contents of IPv6 packets. **Caution** Using this command could flood the system and increase CPU usage if several packets are being encrypted.

• Examples, page 386

Examples

Sample Output from the show crypto ipsec sa Command

The following is sample output from the **show crypto ipsec sa**command:

```
Router# show crypto ipsec sa
interface: Tunnel0
     Crypto map tag: Tunnel0-head-0, local addr 3FFE:2002::A8BB:CCFF:FE01:9002
    protected vrf: (none)
    local  ident (addr/mask/prot/port): (::/0/0/0)
    remote ident (addr/mask/prot/port): (::/0/0/0)
    current_peer 3FFE:2002::A8BB:CCFF:FE01:2C02 port 500
      PERMIT, flags={origin_is_acl,}
     #pkts encaps: 133, #pkts encrypt: 133, #pkts digest: 133
     #pkts decaps: 133, #pkts decrypt: 133, #pkts verify: 133
     #pkts compressed: 0, #pkts decompressed: 0
     #pkts not compressed: 0, #pkts compr. failed: 0
     #pkts not decompressed: 0, #pkts decompress failed: 0
     #send errors 60, #recv errors 0
```

```
local crypto endpt.: 3FFE:2002::A8BB:CCFF:FE01:9002,
remote crypto endpt.: 3FFE:2002::A8BB:CCFF:FE01:2C02
path mtu 1514, ip mtu 1514
current outbound spi: 0x28551D9A(676666778)
inbound esp sas:
 spi: 0x2104850C(553944332)
   transform: esp-des ,
   in use settings ={Tunnel, }
   conn id: 93, flow_id: SW:93, crypto map: Tunnel0-head-0
   sa timing: remaining key lifetime (k/sec): (4397507/148)
   IV size: 8 bytes
   replay detection support: Y
   Status: ACTIVE
inbound ah sas:
 spi: 0x967698CB(2524354763)
   transform: ah-sha-hmac ,
   in use settings ={Tunnel, }
   conn id: 93, flow_id: SW:93, crypto map: Tunnel0-head-0
   sa timing: remaining key lifetime (k/sec): (4397507/147)
   replay detection support: Y
   Status: ACTIVE
inbound pcp sas:
outbound esp sas:
 spi: 0x28551D9A(676666778)
   transform: esp-des ,
   in use settings ={Tunnel, }
   conn id: 94, flow_id: SW:94, crypto map: Tunnel0-head-0
   sa timing: remaining key lifetime (k/sec): (4397508/147)
   IV size: 8 bytes
   replay detection support: Y
   Status: ACTIVE
outbound ah sas:
 spi: 0xA83E05B5(2822636981)
   transform: ah-sha-hmac ,
   in use settings ={Tunnel, }
   conn id: 94, flow_id: SW:94, crypto map: Tunnel0-head-0
   sa timing: remaining key lifetime (k/sec): (4397508/147)
   replay detection support: Y
   Status: ACTIVE
outbound pcp sas:
```

Sample Output from the show crypto isakmp peer Command

The following sample output shows peer descriptions on an IPv6 router:

```
Router# show crypto isakmp peer detail
Peer: 2001:DB8:0:1::1 Port: 500 Local: 2001:DB8:0:2::1
Phase1 id: 2001:DB8:0:1::1
flags:
NAS Port: 0 (Normal)
IKE SAs: 1 IPsec SA bundles: 1
last_locker: 0x141A188, last_last_locker: 0x0
last_unlocker: 0x0, last_last_unlocker: 0x0
```

Sample Output from the show crypto isakmp profile Command

The following sample output shows the ISAKMP profiles that are defined on an IPv6 router.

```
Router# show crypto isakmp profile
ISAKMP PROFILE tom
Identities matched are:
ipv6-address 2001:DB8:0:1::1/32
Certificate maps matched are:
Identity presented is: ipv6-address fqdn
keyring(s): <none>
trustpoint(s): <all>
```

Sample Output from the show crypto isakmp sa Command

The following sample output shows the SAs of an active IPv6 device. The IPv4 device is inactive:

```
Router# show crypto isakmp sa detail

Codes: C - IKE configuration mode, D - Dead Peer Detection
        K - Keepalives, N - NAT-traversal
        X - IKE Extended Authentication
        psk - Preshared key, rsig - RSA signature
        renc - RSA encryption
IPv4 Crypto ISAKMP SA

C-id  Local           Remote          I-VRF    Status Encr Hash Auth DH
Lifetime Cap.

IPv6 Crypto ISAKMP SA

  dst: 3FFE:2002::A8BB:CCFF:FE01:2C02
  src: 3FFE:2002::A8BB:CCFF:FE01:9002
  conn-id: 1001  I-VRF:          Status: ACTIVE Encr: des  Hash: sha  Auth:
psk
  DH: 1  Lifetime: 23:45:00 Cap: D    Engine-id:Conn-id = SW:1

  dst: 3FFE:2002::A8BB:CCFF:FE01:2C02
  src: 3FFE:2002::A8BB:CCFF:FE01:9002
  conn-id: 1002  I-VRF:          Status: ACTIVE Encr: des  Hash: sha  Auth: psk
  DH: 1  Lifetime: 23:45:01 Cap: D    Engine-id:Conn-id = SW:2
```

Sample Output from the show crypto map Command

The following sample output shows the dynamically generated crypto maps of an active IPv6 device:

```
Router# show crypto map

Crypto Map "Tunnel1-head-0" 65536 ipsec-isakmp
        Profile name: profile0
        Security association lifetime: 4608000 kilobytes/300 seconds
        PFS (Y/N): N
```

```
        Transform sets={
                ts,
        }
Crypto Map "Tunnel1-head-0" 65537
        Map is a PROFILE INSTANCE.
        Peer = 2001:1::2
IPv6 access list Tunnel1-head-0-ACL (crypto)
    permit ipv6 any any (61445999 matches) sequence 1
        Current peer: 2001:1::2
        Security association lifetime: 4608000 kilobytes/300 seconds
        PFS (Y/N): N
        Transform sets={
          ts,
        }
        Interfaces using crypto map Tunnel1-head-0:
        Tunnel1
```

Sample Output from the show crypto session Command

The following output from the show crypto session information provides details on currently active crypto sessions:

```
Router# show crypto session detail
Crypto session current status
Code: C - IKE Configuration mode, D - Dead Peer Detection K - Keepalives, N - NAT-
traversal, X - IKE Extended Authentication
Interface: Tunnel1
Session status: UP-ACTIVE
Peer: 2001:1::1 port 500 fvrf: (none) ivrf: (none)
       Phase1_id: 2001:1::1
       Desc: (none)
   IKE SA: local 2001:1::2/500
           remote 2001:1::1/500 Active
           Capabilities:(none) connid:14001 lifetime:00:04:32
   IPSEC FLOW: permit ipv6 ::/0 ::/0
         Active SAs: 4, origin: crypto map
         Inbound:  #pkts dec'ed 42641 drop 0 life (KB/Sec) 4534375/72
         Outbound: #pkts enc'ed 6734980 drop 0 life (KB/Sec) 2392402/72
```

Configuration Examples for IPsec for IPv6 Security

Example: Configuring a VTI for Site-to-Site IPv6 IPsec Protection

```
crypto isakmp policy 1
  authentication pre-share
!
crypto isakmp key myPreshareKey0 address ipv6 3FFE:2002::A8BB:CCFF:FE01:2C02/128
crypto isakmp keepalive 30 30
!
crypto ipsec transform-set 3des ah-sha-hmac esp-3des
!
crypto ipsec profile profile0
  set transform-set 3des
!
ipv6 cef
!
interface Tunnel0
  ipv6 address 3FFE:1001::/64 eui-64
  ipv6 enable
  ipv6 cef
  tunnel source Ethernet2/0
  tunnel destination 3FFE:2002::A8BB:CCFF:FE01:2C02
```

```
tunnel mode ipsec ipv6
tunnel protection ipsec profile profile0
```

Additional References

Related Documents

Related Topic	Document Title
OSPFv3 authentication support with IPsec	Implementing OSPFv3
IPsec VTI information	IPsec Virtual Tunnel Interface
IPv6 supported feature list	Start Here: Cisco IOS Software Release Specifics for IPv6 Features
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*
IPv4 security configuration tasks	*Cisco IOS Security Configuration Guide*
IPv4 security commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS Security Command Reference*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIBs	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 2401	*Security Architecture for the Internet Protocol*

RFCs	Title
RFC 2402	*IP Authentication Header*
RFC 2404	*The Use of Hash Message Authentication Code Federal Information Processing Standard 180-1 within Encapsulating Security Payload and Authentication Header*
RFC 2406	*IP Encapsulating Security Payload (ESP)*
RFC 2407	*The Internet Security Domain of Interpretation for ISAKMP*
RFC 2408	*Internet Security Association and Key Management Protocol (ISAKMP)*
RFC 2409	*Internet Key Exchange (IKE)*
RFC 2460	*Internet Protocol, Version 6 (IPv6) Specification*
RFC 2474	*Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*
RFC 3576	*Change of Authorization*
RFC 4109	*Algorithms for Internet Key Exchange version 1 (IKEv1)*
RFC 4302	*IP Authentication Header*
RFC 4306	*Internet Key Exchange (IKEv2) Protocol*
RFC 4308	*Cryptographic Suites for IPsec*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing IPsec in IPv6 Security

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software

release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 18 Feature Information for Implementing IPsec in IPv6 Security

Feature Name	Releases	Feature Information
IPv6 IPsec to Authenticate Open Shortest Path First for IPv6 (OSPFv3)	12.3(4)T 12.4 12.4(2)T	OSPF for IPv6 uses the IPsec secure socket application program interface (API) to add authentication to OSPF for IPv6 packets. This API has been extended to provide support for IPv6.
IPv6 IPsec VPN	12.4(4)T	
IPsec IPv6 Phase 2 Support	12.4(4)T	Features in this phase support tunnel mode for site-to-site IPsec protection of IPv6 traffic. This feature allows the use of IPv6 IPsec encapsulation to protect IPv6 unicast and multicast traffic.

Implementing IS-IS for IPv6

This module describes how to configure Integrated Intermediate System-to-Intermediate System (IS-IS) for IPv6. IS-IS is an Interior Gateway Protocol (IGP) that advertises link-state information throughout the network to create a picture of the network topology. IS-IS is an Open Systems Interconnection (OSI) hierarchical routing protocol that designates an intermediate system as a Level 1 or Level 2 device. Level 2 devices route between Level 1 areas to create an intradomain routing backbone. Integrated IS-IS uses a single routing algorithm to support several network address families, such as IPv6, IPv4, and OSI.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Implementing IS-IS for IPv6

In Cisco IOS Release 12.0(22)S or later releases, and Cisco IOS Release 12.2(8)T or later releases, IS-IS support for IPv6 implements single-topology IPv6 IS-IS functionality based on IETF IS-IS WG draft-ietf-isis-ipv6.txt. A single shortest path first (SPF) per level is used to compute OSI, IPv4 (if configured), and IPv6 routes. The use of a single SPF means that both IPv4 IS-IS and IPv6 IS-IS routing protocols must share a common network topology. To use IS-IS for IPv4 and IPv6 routing, any interface configured for IPv4 IS-IS must also be configured for IPv6 IS-IS, and vice versa. All routers within an IS-IS area (Level 1 routing) or domain (Level 2 routing) must also support the same set of address families: IPv4 only, IPv6 only, or both IPv4 and IPv6.

Beginning with release Cisco IOS Release 12.2(15)T, IS-IS support for IPv6 is enhanced to also support multitopology IPv6 support as defined in IETF IS-IS WG draft-ietf-isis-wg-multi-topology.txt.

Multitopology IPv6 IS-IS support uses multiple SPFs to compute routes and removes the restriction that all interfaces must support all configured address families and that all routers in an IS-IS area or domain must support the same set of address families.

The following IS-IS router configuration commands are specific to IPv4 and are not supported by, or have any effect on, IPv6 IS-IS:

- **mpls**
- **traffic-share**

Information About Implementing IS-IS for IPv6

IS-IS Enhancements for IPv6

IS-IS in IPv6 functions the same and offers many of the same benefits as IS-IS in IPv4. IPv6 enhancements to IS-IS allow IS-IS to advertise IPv6 prefixes in addition to IPv4 and OSI routes. Extensions to the IS-IS command-line interface (CLI) allow configuration of IPv6-specific parameters. IPv6 IS-IS extends the address families supported by IS-IS to include IPv6, in addition to OSI and IPv4.

IS-IS in IPv6 supports either single-topology mode or multiple topology mode.

IS-IS Single-Topology Support for IPv6

Single-topology support for IPv6 allows IS-IS for IPv6 to be configured on interfaces along with other network protocols (for example, IPv4 and Connectionless Network Service [CLNS]). All interfaces must be configured with the identical set of network address families. In addition, 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 address families on all interfaces.

When single-topology support for IPv6 is being used, either old- or new-style TLVs may be used. However, the TLVs used to advertise reachability to IPv6 prefixes use extended metrics. Cisco routers do not allow an interface metric to be set to a value greater than 63 if the configuration is not set to support only new-style TLVs for IPv4. In single-topology IPv6 mode, the configured metric is always the same for both IPv4 and IPv6.

IS-IS Multitopology Support for IPv6

IS-IS multitopology support for IPv6 allows IS-IS to maintain a set of independent topologies within a single area or domain. This mode removes the restriction that all interfaces on which IS-IS is configured must support the identical set of network address families. It also removes the restriction that all routers in the IS-IS area (for Level 1 routing) or domain (for Level 2 routing) must support the identical set of network layer address families. Because multiple SPFs are performed, one for each configured topology, it is sufficient that connectivity exists among a subset of the routers in the area or domain for a given network address family to be routable.

You can use the **isis ipv6 metric** command to configure different metrics on an interface for IPv6 and IPv4.

When multitopology support for IPv6 is used, use the **metric-style wide**command to configure IS-IS to use new-style TLVs because TLVs used to advertise IPv6 information in link-state packets (LSPs) are defined to use only extended metrics.

Transition from Single-Topology to Multitopology Support for IPv6

All routers in the area or domain must use the same type of IPv6 support, either single-topology or multitopology. A router operating in multitopology mode will not recognize the ability of the single-topology mode router to support IPv6 traffic, which will lead to holes in the IPv6 topology. To transition from single-topology support to the more flexible multitopology support, a multitopology transition mode is provided.

The multitopology transition mode allows a network operating in single-topology IS-IS IPv6 support mode to continue to work while upgrading routers to include multitopology IS-IS IPv6 support. While in transition mode, both types of TLVs (single-topology and multitopology) are sent in LSPs for all configured IPv6 addresses, but the router continues to operate in single-topology mode (that is, the topological restrictions of the single-topology mode are still in effect). After all routers in the area or domain have been upgraded to support multitopology IPv6 and are operating in transition mode, transition mode can be removed from the configuration. Once all routers in the area or domain are operating in multitopology IPv6 mode, the topological restrictions of single-topology mode are no longer in effect.

IPv6 IS-IS Local RIB

A router that is running IS-IS IPv6 maintains a local RIB in which it stores all routes to destinations it has learned from its neighbors. At the end of each SPF, IS-IS attempts to install the best (that is, the least-cost) routes to a destination present in the local RIB in the global IPv6 routing table.

How to Implement IS-IS for IPv6

When configuring supported routing protocols in IPv6, you must create the routing process, enable the routing process on interfaces, and customize the routing protocol for your particular network.

- Configuring Single-Topology IS-IS for IPv6, page 395
- Configuring Multitopology IS-IS for IPv6, page 397
- Customizing IPv6 IS-IS, page 399
- Redistributing Routes into an IPv6 IS-IS Routing Process, page 402
- Redistributing IPv6 IS-IS Routes Between IS-IS Levels, page 403
- Disabling IPv6 Protocol-Support Consistency Checks, page 404
- Disabling IPv4 Subnet Consistency Checks, page 405
- Verifying IPv6 IS-IS Configuration and Operation, page 406

Configuring Single-Topology IS-IS for IPv6

Perform this task to create an IPv6 IS-IS process and enable IPv6 IS-IS support on an interface.

Configuring IS-IS comprises two activities. The first activity creates an IS-IS routing process and is performed using protocol-independent IS-IS commands. The second activity in configuring IPv6 IS-IS configures the operation of the IS-IS protocol on an interface.

Before configuring the router to run IPv6 IS-IS, globally enable IPv6 using the **ipv6 unicast-routing** global configuration command.

Note If you are using IS-IS single-topology support for IPv6, IPv4, or both IPv6 and IPv4, you may configure both IPv6 and IPv4 on an IS-IS interface for Level 1, Level 2, or both Level 1 and Level 2. However, if both IPv6 and IPv4 are configured on the same interface, they must be running the same IS-IS level. That is, IPv4 cannot be configured to run on IS-IS Level 1 only on a specified Ethernet interface while IPv6 is configured to run IS-IS Level 2 only on the same Ethernet interface.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router isis** *area-tag*
4. **net** *network-entity-title*
5. **exit**
6. **interface** *type number*
7. **ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits* / *prefix-length*
8. **ipv6 router isis** *area-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router isis** *area-tag* **Example:** `Router(config)# router isis area2`	Enables IS-IS for the specified IS-IS routing process, and enters router configuration mode.

	Command or Action	Purpose
Step 4	**net** *network-entity-title* **Example:** `Router(config-router)# net 49.0001.0000.0000.000c.00`	Configures an IS-IS network entity title (NET) for the routing process. • The *network-entity-title* argument defines the area addresses for the IS-IS area and the system ID of the router. **Note** For more details about the format of the *network-entity-title* argument, refer to the " Configuring ISO CLNS " chapter in the *Cisco IOS ISO CLNS Configuration Guide* .
Step 5	**exit** **Example:** `Router(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 6	**interface** *type number* **Example:** `Router(config)# interface Ethernet 0/0/1`	Specifies the interface type and number, and enters interface configuration mode.
Step 7	**ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits* / *prefix-length* **Example:** `Router(config-if)# ipv6 address 2001:DB8::3/64`	Specifies the IPv6 network assigned to the interface and enables IPv6 processing on the interface. **Note** Refer to Implementing IPv6 Addressing and Basic Connectivity for more information on configuring IPv6 addresses.
Step 8	**ipv6 router isis** *area-name* **Example:** `Router(config-if)# ipv6 router isis area2`	Enables the specified IPv6 IS-IS routing process on an interface.

Configuring Multitopology IS-IS for IPv6

When multitopology IS-IS for IPv6 is configured, the **transition** keyword allows a user who is working with the single-topology SPF mode of IS-IS IPv6 to continue to work while upgrading to multitopology IS-IS. After every router is configured with the **transition** keyword, users can remove the **transition** keyword on each router. When transition mode is not enabled, IPv6 connectivity between routers operating in single-topology mode and routers operating in multitopology mode is not possible.

You can continue to use the existing IPv6 topology while upgrading to multitopology IS-IS. The optional **isis ipv6 metric** command allows you to differentiate between link costs for IPv6 and IPv4 traffic when operating in multitopology mode.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router isis** *area-tag*
4. **metric-style wide** [**transition**] [**level-1** | **level-2** | **level-1-2**
5. **address-family ipv6** [**unicast** | **multicast**]
6. **multi-topology** [**transition**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router isis** *area-tag* **Example:** `Router(config)# router isis area2`	Enables IS-IS for the specified IS-IS routing process, and enters router configuration mode.
Step 4	**metric-style wide** [**transition**] [**level-1**	**level-2**	**level-1-2** **Example:** `Router(config-router)# metric-style wide level-1`	Configures a router running IS-IS to generate and accept only new-style TLVs.
Step 5	**address-family ipv6** [**unicast**	**multicast**] **Example:** `Router(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the unicast IPv6 unicast address family. By default, the router is placed in configuration mode for the unicast IPv6 address family if the **unicast** keyword is not specified with the **address-family ipv6** command.

	Command or Action	Purpose
Step 6	**multi-topology** [**transition**] **Example:** `Router(config-router-af)# multi-topology`	Enables multitopology IS-IS for IPv6. • The optional **transition** keyword allows an IS-IS IPv6 user to continue to use single-topology mode while upgrading to multitopology mode.

Customizing IPv6 IS-IS

Perform this task to configure a new administrative distance for IPv6 IS-IS, configure the maximum number of equal-cost paths that IPv6 IS-IS will support, configure summary prefixes for IPv6 IS-IS, and configure an IS-IS instance to advertise the default IPv6 route (::/0). It also explains how to configure the hold-down period between partial route calculations (PRCs) and how often Cisco IOS software performs the SPF calculation when using multitopology IS-IS.

You can customize IS-IS multitopology for IPv6 for your network, but you likely will not need to do so. The defaults for this feature are set to meet the requirements of most customers and features. If you change the defaults, refer to the IPv4 configuration guide and the IPv6 command reference to find the appropriate syntax.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router isis** *area-tag*
4. **address-family ipv6** [**unicast** | **multicast**]
5. **default-information originate** [**route-map** *map-name*]
6. **distance** *value*
7. **maximum-paths** *number-paths*
8. **summary-prefix** *ipv6-prefix prefix-length* [**level-1** | **level-1-2**| **level-2**]
9. **prc-interval** *seconds* [*initial-wait*] [*secondary-wait*]
10. **spf-interval** [**level-1** | **level-2**] *seconds initial-wait*] [*secondary-wait*]
11. **exit**
12. **interface** *type number*
13. **isis ipv6 metric** *metric-value* [**level-1** | **level-2** | **level-1-2**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router isis** *area-tag* **Example:** `Router(config)# router isis area2`	Enables IS-IS for the specified IS-IS routing process, and enters router configuration mode.
Step 4	**address-family ipv6** [**unicast** | **multicast**] **Example:** `Router(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the unicast IPv6 unicast address family. By default, the router is placed in configuration mode for the unicast IPv6 address family if the **unicast** keyword is not specified with the **address-family ipv6** command.
Step 5	**default-information originate** [**route-map** *map-name*] **Example:** `Router(config-router-af)# default-information originate`	(Optional) Injects a default IPv6 route into an IS-IS routing domain. • The **route-map** keyword and *map-name* argument specify the conditions under which the IPv6 default route is advertised. • If the **route map** keyword is omitted, then the IPv6 default route will be unconditionally advertised at Level 2.
Step 6	**distance** *value* **Example:** `Router(config-router-af)# distance 90`	(Optional) Defines an administrative distance for IPv6 IS-IS routes in the IPv6 routing table. • The *value*argument is an integer from 10 to 254. (The values 0 to 9 are reserved for internal use).
Step 7	**maximum-paths** *number-paths* **Example:** `Router(config-router-af)# maximum-paths 3`	(Optional) Defines the maximum number of equal-cost routes that IPv6 IS-IS can support. • This command also supports IPv6 Border Gateway Protocol (BGP) and Routing Information Protocol (RIP). • The *number-paths* argument is an integer from 1 to 64. The default for BGP is one path; the default for IS-IS and RIP is 16 paths.

	Command or Action	Purpose
Step 8	**summary-prefix** *ipv6-prefix prefix-length* [**level-1** | **level-1-2**| **level-2**] **Example:** `Router(config-router-af)# summary-prefix 2001:DB8::/24`	(Optional) Allows a Level 1-2 router to summarize Level 1 prefixes at Level 2, instead of advertising the Level 1 prefixes directly when the router advertises the summary. • The *ipv6-prefix* argument in the **summary-prefix** command must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons. • The *prefix-length* argument is a decimal value that indicates how many of the high-order contiguous bits of the address comprise the prefix (the network portion of the address). A slash mark must precede the decimal value.
Step 9	**prc-interval** *seconds* [*initial-wait*] [*secondary-wait*] **Example:** `Router(config-router-af)# prc-interval 20`	(Optional) Configures the hold-down period between PRCs for multitopology IS-IS for IPv6.
Step 10	**spf-interval** [**level-1** | **level-2**] *seconds initial-wait*] [*secondary-wait*] **Example:** `Router(config-router-af)# spf-interval 30`	(Optional) Configures how often Cisco IOS software performs the SPF calculation for multitopology IS-IS for IPv6.
Step 11	**exit** **Example:** `Router(config-router-af)# exit`	Exits address family configuration mode, and returns the router to router configuration mode. • Repeat this step to exit router configuration mode and return the router to global configuration mode.
Step 12	**interface** *type number* **Example:** `Router(config-router)# interface Ethernet 0`	Specifies the interface type and number, and enters interface configuration mode.
Step 13	**isis ipv6 metric** *metric-value* [**level-1** | **level-2** | **level-1-2**] **Example:** `Router(config-if)# isis ipv6 metric 20`	(Optional) Configures the value of an multitopology IS-IS for IPv6 metric.

Redistributing Routes into an IPv6 IS-IS Routing Process

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router isis** *area-tag*
4. **address-family ipv6** [**unicast** | **multicast**]
5. **redistribute** *source-protocol process-id*] [**include-connected**] [*target-protocol-options*] [*source-protocol-options*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router isis** *area-tag* **Example:** `Router(config)# router isis area2`	Enables IS-IS for the specified IS-IS routing process, and enters router configuration mode.
Step 4	**address-family ipv6** [**unicast**	**multicast**] **Example:** `Router(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the unicast IPv6 unicast address family. By default, the router is placed in configuration mode for the unicast IPv6 address family if the **unicast** keyword is not specified with the **address-family ipv6** command.

	Command or Action	Purpose
Step 5	**redistribute** *source-protocol process-id*] [**include-connected**] [*target-protocol-options*] [*source-protocol-options*] **Example:** `Router(config-router-af)# redistribute bgp 64500 metric 100 route-map isismap`	Redistributes routes from the specified protocol into the IS-IS process. • The *source-protocol* argument can be one of the following keywords: **bgp**, **connected**, **isis**, **rip**, or **static**. • Only the arguments and keywords relevant to this task are specified here.

Redistributing IPv6 IS-IS Routes Between IS-IS Levels

Perform this task to redistribute IPv6 routes learned at one IS-IS level into a different level.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router isis** *area-tag*
4. **address-family ipv6** [**unicast** | **multicast**]
5. **redistribute isis** [*process-id*] {**level-1** | **level-2**} **into** {**level-1** | **level-2**} **distribute-list** *list-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router isis** *area-tag* **Example:** `Router(config)# router isis area2`	Enables IS-IS for the specified IS-IS routing process, and enters router configuration mode.

	Command or Action	Purpose
Step 4	**address-family ipv6** [**unicast** | **multicast**] **Example:** `Router(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the unicast IPv6 unicast address family. By default, the router is placed in configuration mode for the unicast IPv6 address family if the **unicast** keyword is not specified with the **address-family ipv6** command.
Step 5	**redistribute isis** [*process-id*] {**level-1** | **level-2**} **into** {**level-1** | **level-2**} **distribute-list** *list-name* **Example:** `Router(config-router-af)# redistribute isis level-1 into level-2`	Redistributes IPv6 routes from one IS-IS level into another IS-IS level. • By default, the routes learned by Level 1 instances are redistributed by the Level 2 instance. **Note** The *protocol* argument must be **isis** in this configuration of the **redistribute** command. Only the arguments and keywords relevant to this task are specified here.

Disabling IPv6 Protocol-Support Consistency Checks

Perform this task to disable protocol-support consistency checks in IPv6 single-topology mode.

For single-topology IS-IS IPv6, routers must be configured to run the same set of address families. IS-IS performs consistency checks on hello packets and will reject hello packets that do not have the same set of configured address families. For example, a router running IS-IS for both IPv4 and IPv6 will not form an adjacency with a router running IS-IS for IPv4 or IPv6 only. In order to allow adjacency to be formed in mismatched address-families network, the **adjacency-check** command in IPv6 address family configuration mode must be disabled.

Note Entering the **no adjacency-check** command can adversely affect your network configuration. Enter the **no adjacency-check** command only when you are running IPv4 IS-IS on all your routers and you want to add IPv6 IS-IS to your network but you need to maintain all your adjacencies during the transition. When the IPv6 IS-IS configuration is complete, remove the **no adjacency-check**command from the configuration.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router isis** *area-tag*
4. **address-family ipv6** [**unicast** | **multicast**]
5. **no adjacency-check**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router isis** *area-tag* **Example:** `Router(config)# router isis area2`	Enables IS-IS for the specified IS-IS routing process, and enters router configuration mode.
Step 4	**address-family ipv6** [**unicast** | **multicast**] **Example:** `Router(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the unicast IPv6 unicast address family. By default, the router is placed in configuration mode for the unicast IPv6 address family if the **unicast** keyword is not specified with the **address-family ipv6** command.
Step 5	**no adjacency-check** **Example:** `Router(config-router-af)# no adjacency-check`	Disables the IPv6 protocol-support consistency checks performed on hello packets, allowing IPv6 to be introduced into an IPv4-only network without disrupting existing adjacencies. • The **adjacency-check**command is enabled by default.

Disabling IPv4 Subnet Consistency Checks

Perform this task to disable IPv4 subnet consistency checking when forming adjacencies. Cisco IOS software historically makes checks on hello packets to ensure that the IPv4 address is present and has a consistent subnet with the neighbor from which the hello packets are received. To disable this check, use the **no adjacency-check** command in the router configuration mode. However, if multitopology IS-IS is configured, this check is automatically suppressed, because multitopology IS-IS requires routers to form an adjacency regardless of whether or not all routers on a LAN support a common protocol.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router isis** *area-tag*
4. **no adjacency-check**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router isis** *area-tag* **Example:** `Router(config)# router isis area2`	Enables IS-IS for the specified IS-IS routing process, and enters router configuration mode.
Step 4	**no adjacency-check** **Example:** `Router(config-router-af)# no adjacency-check`	Disables the IPv6 protocol-support consistency checks performed on hello packets, allowing IPv6 to be introduced into an IPv4-only network without disrupting existing adjacencies. • The **adjacency-check**command is enabled by default.

Verifying IPv6 IS-IS Configuration and Operation

SUMMARY STEPS

1. **enable**
2. **show ipv6 protocols** [**summary**]
3. **show isis** [*process-tag*] [**ipv6** | *****] **topology**
4. **show clns** [*process-tag*] **neighbors** *interface-type interface-number*] [**area**] [**detail**]
5. **show clns** *area-tag* **is-neighbors** [*type number*] [**detail**]
6. **show isis** [*process-tag*] **database** [**level-1**] [**level-2**] [**l1**] [**l2**] [**detail**] [**lspid**]
7. **show isis ipv6 rib** [*ipv6-prefix*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ipv6 protocols** [**summary**] **Example:** `Router# show ipv6 protocols`	Displays the parameters and current state of the active IPv6 routing processes.
Step 3	**show isis** [*process-tag*] [**ipv6** | *****] **topology** **Example:** `Router# show isis topology`	Displays a list of all connected routers running IS-IS in all areas.
Step 4	**show clns** [*process-tag*] **neighbors** *interface-type interface-number*] [**area**] [**detail**] **Example:** `Router# show clns neighbors detail`	Displays end system (ES), intermediate system (IS), and multitopology IS-IS (M-ISIS) neighbors.
Step 5	**show clns** *area-tag* **is-neighbors** [*type number*] [**detail**] **Example:** `Router# show clns is-neighbors detail`	Displays IS-IS adjacency information for IS-IS neighbors. • Use the **detail** keyword to display the IPv6 link-local addresses of the neighbors.
Step 6	**show isis** [*process-tag*] **database** [**level-1**] [**level-2**] [**l1**] [**l2**] [**detail**] [**lspid**] **Example:** `Router# show isis database detail`	Displays the IS-IS link-state database. • In this example, the contents of each LSP are displayed using the **detail** keyword.
Step 7	**show isis ipv6 rib** [*ipv6-prefix*] **Example:** `Router# show isis ipv6 rib`	Displays the IPv6 local RIB.

Examples

Sample Output from the show ipv6 protocols Command

In the following example, output information about the parameters and current state of that active IPv6 routing processes is displayed using the **show ipv6 protocols**command:

```
Router# show ipv6 protocols
IPv6 Routing Protocol is "connected"
IPv6 Routing Protocol is "static"
IPv6 Routing Protocol is "isis"
  Interfaces:
    Ethernet0/0/3
    Ethernet0/0/1
    Serial1/0/1
    Loopback1 (Passive)
    Loopback2 (Passive)
    Loopback3 (Passive)
    Loopback4 (Passive)
    Loopback5 (Passive)
  Redistribution:
    Redistributing protocol static at level 1
  Address Summarization:
    L2: 2001:DB8:33::/16  advertised with metric 0
    L2: 2001:DB8:44::/16  advertised with metric 20
    L2: 2001:DB8:66::/16  advertised with metric 10
    L2: 2001:DB8:77::/16  advertised with metric 10
```

Sample Output from the show isis topology Command

In the following example, output information about all connected routers running IS-IS in all areas is displayed using the **show isis topology**command:

```
Router# show isis topology
IS-IS paths to level-1 routers
System Id       Metric  Next-Hop        Interface       SNPA
0000.0000.000C
0000.0000.000D  20      0000.0000.00AA  Se1/0/1         *HDLC*
0000.0000.000F  10      0000.0000.000F  Et0/0/1         0050.e2e5.d01d
0000.0000.00AA  10      0000.0000.00AA  Se1/0/1         *HDLC*
IS-IS paths to level-2 routers
System Id       Metric  Next-Hop        Interface       SNPA
0000.0000.000A  10      0000.0000.000A  Et0/0/3         0010.f68d.f063
0000.0000.000B  20      0000.0000.000A  Et0/0/3         0010.f68d.f063
0000.0000.000C  --
0000.0000.000D  30      0000.0000.000A  Et0/0/3         0010.f68d.f063
0000.0000.000E  30      0000.0000.000A  Et0/0/3         0010.f68d.f063
```

Sample Output from the show clns is-neighbors Command

In the following example, output information to confirm that the local router has formed all the necessary IS-IS adjacencies with other IS-IS neighbors is displayed using the **show clns is-neighbors**command. To display the IPv6 link-local addresses of the neighbors, specify the **detail** keyword.

```
Router# show clns is-neighbors detail
System Id      Interface   State  Type Priority  Circuit Id         Format
0000.0000.00AA Se1/0/1     Up     L1   0         00                 Phase V
  Area Address(es): 49.0001
  IPv6 Address(es): FE80::YYYY:D37C:C854:5
  Uptime: 17:21:38
0000.0000.000F Et0/0/1     Up     L1   64        0000.0000.000C.02  Phase V
  Area Address(es): 49.0001
```

```
  IPv6 Address(es): FE80::XXXX:E2FF:FEE5:D01D
  Uptime: 17:21:41
0000.0000.000A Et0/0/3     Up     L2   64        0000.0000.000C.01  Phase V
  Area Address(es): 49.000b
  IPv6 Address(es): FE80::ZZZZ:F6FF:FE8D:F063
  Uptime: 17:22:06
```

Sample Output from the show clns neighbors Command

In the following example, detailed output information that displays both end system (ES) and intermediate system (IS) neighbors is displayed using the **show clns neighbors** command with the **detail** keyword.

```
Router# show clns neighbors detail
System Id          Interface    SNPA            State  Holdtime  Type Protocol
0000.0000.0007     Et3/3        aa00.0400.6408  UP     26        L1   IS-IS
Area Address(es): 20
IP Address(es): 172.16.0.42*
Uptime: 00:21:49
0000.0C00.0C35     Et3/2        0000.0c00.0c36  Up     91        L1   IS-IS
Area Address(es): 20
IP Address(es): 192.168.0.42*
Uptime: 00:21:52
0800.2B16.24EA     Et3/3        aa00.0400.2d05  Up     27        L1   M-ISIS
Area Address(es): 20
IP Address(es): 192.168.0.42*
IPv6 Address(es): FE80::2B0:8EFF:FE31:EC57
Uptime: 00:00:27
0800.2B14.060E     Et3/2        aa00.0400.9205  Up     8         L1   IS-IS
Area Address(es): 20
IP Address(es): 192.168.0.30*
Uptime: 00:21:52
```

Sample Output from the show isis database Command

In the following example, detailed output information about LSPs received from other routers and the IPv6 prefixes they are advertising is displayed using the **show isis database**command with the **detail** keyword specified:

```
Router# show isis database detail
IS-IS Level-1 Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime  ATT/P/OL
0000.0C00.0C35.00-00  0x0000000C   0x5696        325           0/0/0
  Area Address: 47.0004.004D.0001
  Area Address: 39.0001
  Metric: 10   IS 0000.0C00.62E6.03
  Metric: 0    ES 0000.0C00.0C35
 --More--
0000.0C00.40AF.00-00* 0x00000009   0x8452        608           1/0/0
  Area Address: 47.0004.004D.0001
  Topology: IPv4 (0x0) IPv6 (0x2)
  NLPID: 0xCC 0x8E
  IP Address: 172.16.21.49
  Metric: 10   IS 0800.2B16.24EA.01
  Metric: 10   IS 0000.0C00.62E6.03
  Metric: 0    ES 0000.0C00.40AF
  IPv6 Address: 2001:DB8::/32
  Metric: 10   IPv6 (MT-IPv6) 2001:DB8::/64
  Metric: 5    IS-Extended cisco.03
  Metric: 10   IS-Extended cisco1.03
  Metric: 10    IS (MT-IPv6) cisco.03
IS-IS Level-2 Link State Database:
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime      ATT/P/OL
0000.0000.000A.00-00  0x00000059   0x378A        949               0/0/0
  Area Address: 49.000b
  NLPID:        0x8E
  IPv6 Address: 2001:DB8:1:1:1:1:1:1
  Metric: 10         IPv6 2001:DB8:2:YYYY::/64
  Metric: 10         IPv6 2001:DB8:3:YYYY::/64
  Metric: 10         IPv6 2001:DB8:2:YYYY::/64
```

```
  Metric: 10         IS-Extended 0000.0000.000A.01
  Metric: 10         IS-Extended 0000.0000.000B.00
  Metric: 10         IS-Extended 0000.0000.000C.01
  Metric: 0          IPv6 11:1:YYYY:1:1:1:1:1/128
  Metric: 0          IPv6 11:2:YYYY:1:1:1:1:1/128
  Metric: 0          IPv6 11:3:YYYY:1:1:1:1:1/128
  Metric: 0          IPv6 11:4:YYYY:1:1:1:1:1/128
  Metric: 0          IPv6 11:5:YYYY:1:1:1:1:1/128
0000.0000.000A.01-00  0x00000050   0xB0AF        491               0/0/0
  Metric: 0          IS-Extended 0000.0000.000A.00
  Metric: 0          IS-Extended 0000.0000.000B.00
```

Sample Output from the show isis ipv6 rib Command

The following example shows output from the **show isis ipv6 rib** command. An asterisk (*) indicates prefixes that have been installed in the master IPv6 RIB as IS-IS routes. Following each prefix is a list of all paths in order of preference, with optimal paths listed first and suboptimal paths listed after optimal paths.

```
Router# show isis ipv6 rib

IS-IS IPv6 process "", local RIB
  2001:DB8:88:1::/64
    via FE80::210:7BFF:FEC2:ACC9/Ethernet2/0, type L2  metric 20 LSP [3/7]
    via FE80::210:7BFF:FEC2:ACCC/Ethernet2/1, type L2  metric 20 LSP [3/7]
* 2001:DB8:1357:1::/64
    via FE80::202:7DFF:FE1A:9471/Ethernet2/1, type L2  metric 10 LSP [4/9]
* 2001:DB8:45A::/64
    via FE80::210:7BFF:FEC2:ACC9/Ethernet2/0, type L1  metric 20 LSP [C/6]
    via FE80::210:7BFF:FEC2:ACCC/Ethernet2/1, type L1  metric 20 LSP [C/6]
    via FE80::210:7BFF:FEC2:ACC9/Ethernet2/0, type L2  metric 20 LSP [3/7]
    via FE80::210:7BFF:FEC2:ACCC/Ethernet2/1, type L2  metric 20 LSP [3/7]
```

Configuration Examples for IPv6 IS-IS

Example Configuring Single-Topology IS-IS for IPv6

The following example enables single-topology mode, creates an IS-IS process, defines the NET, configures an IPv6 address on an interface, and configures the interface to run IPv6 IS-IS:

```
ipv6 unicast-routing
!
router isis
 net 49.0001.0000.0000.000c.00
 exit
interface Ethernet0/0/1
 ipv6 address 2001:DB8::3/64
 ipv6 router isis area2
```

Example: Customizing IPv6 IS-IS

The following example advertises the IPv6 default route (::/0)--with an origin of Ethernet interface 0/0/1--with all other routes in router updates sent on Ethernet interface 0/0/1. This example also sets an administrative distance for IPv6 IS-IS to 90, defines the maximum number of equal-cost paths that IPv6 IS-IS will support as 3, and configures a summary prefix of 2001:DB8::/24 for IPv6 IS-IS.

```
router isis
 address-family ipv6
 default-information originate
 distance 90
 maximum-paths 3
 summary-prefix 2001:DB8::/24
 exit
```

Example: Redistributing Routes into an IPv6 IS-IS Routing Process

The following example redistributes IPv6 BGP routes into the IPv6 IS-IS Level 2 routing process:

```
router isis
 address-family ipv6
 redistribute bgp 64500 metric 100 route-map isismap
 exit
```

Example: Redistributing IPv6 IS-IS Routes Between IS-IS Levels

The following example redistributes IPv6 IS-IS Level 1 routes into the IPv6 IS-IS Level 2 routing process:

```
router isis
 address-family ipv6
 redistribute isis level-1 into level-2
```

Example: Disabling IPv6 Protocol-Support Consistency Checks

The following example disables the **adjacency-check** command to allow a network administrator to configure IPv6 IS-IS on the router without disrupting the existing adjacencies:

```
router isis
 address-family ipv6
 no adjacency-check
```

Example Configuring Multitopology IS-IS for IPv6

The following example configures multitopology IS-IS in IPv6 after you have configured IS-IS for IPv6:

```
router isis
 metric-style wide
 address-family ipv6
 multi-topology
```

Example: Configuring the IS-IS IPv6 Metric for Multitopology IS-IS

The following example sets the value of an IS-IS IPv6 metric to 20:

```
interface GigabitEthernet 0/0/1
 isis ipv6 metric 20
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 supported feature list	"Start Here: Cisco IOS Software Release Specifics for IPv6 Features ," *Cisco IOS IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*
IS-IS configuration tasks	*Cisco IOS IP Routing Protocols Configuration Guide*
IS-IS commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IP Routing Protocols Command Reference*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIBs	MIBs Link
• CISCO-IETF-IP-FORWARD-MIB • CISCO-IETF-IP-MIB	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 1195	*Use of OSI IS-IS for Routing in TCP/IP and Dual Environments*
RFC 5120	*M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)*
RFC 5308	*Routing IPv6 with IS-IS*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing IS-IS for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 19 Feature Information for Implementing IS-IS for IPv6

Feature Name	Releases	Feature Information
IPv6 Routing--Route Redistribution	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	IS-IS for IPv6 supports redistributing routes into an IPv6 IS-IS routing process and redistributing IPv6 IS-IS routes between IS-IS levels.
IPv6 Routing--IS-IS Support for IPv6	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(8)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	IPv6 enhancements to IS-IS allow IS-IS to advertise IPv6 prefixes in addition to IPv4 and OSI routes.

Feature Name	Releases	Feature Information
IPv6 Routing--IS-IS Multitopology Support for IPv6	12.0(26)S 12.2(18)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA12.2(15)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	IS-IS multitopology support for IPv6 allows IS-IS to maintain a set of independent topologies within a single area or domain.
IPv6 Routing--IS-IS Local RIB	12,2(22)S 12.2(33)SRA 12.2(33)SXH	A router that is running IS-IS IPv6 maintains a local RIB in which it stores all routes to destinations it has learned from its neighbors.

Implementing IPv6 for Network Management

This document describes the concepts and commands used to manage Cisco applications over IPv6 and to implement IPv6 for network management.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Implementing IPv6 for Network Management

Telnet Access over IPv6

The Telnet client and server in Cisco software support IPv6 connections. A user can establish a Telnet session directly to the device using an IPv6 Telnet client, or an IPv6 Telnet connection can be initiated

from the device. A vty interface and password must be created in order to enable Telnet access to an IPv6 device.

TFTP IPv6 Support

TFTP is designed to transfer files over the network from one host to another using the most minimal set of functionality possible. TFTP uses a client/server model in which clients can request to copy files to or from a server. TFTP uses UDP over IPv4 or IPv6 as its transport, and it can work over IPv4 and IPv6 network layers.

- TFTP File Downloading for IPv6, page 416

TFTP File Downloading for IPv6

IPv6 supports TFTP file downloading and uploading using the **copy** command. The **copy** command accepts a destination IPv6 address or IPv6 hostname as an argument and saves the running configuration of the device to an IPv6 TFTP server, as follows:

```
Device# copy running-config tftp://[3ffe:xxxx:c18:1:290:27ff:fe3a:9e9a]/running-config
```

ping and traceroute Commands in IPv6

The **ping** command accepts a destination IPv6 address or IPv6 hostname as an argument and sends Internet Control Message Protocol version 6 (ICMPv6) echo request messages to the specified destination. The ICMPv6 echo reply messages are reported on the console. Extended ping functionality is also supported in IPv6.

The **traceroute** command accepts a destination IPv6 address or IPv6 hostname as an argument and will generate IPv6 traffic to report each IPv6 hop used to reach the destination address.

SSH over an IPv6 Transport

Secure shell (SSH) SSH in IPv6 functions the same and offers the same benefits as SSH in IPv4. The SSH server feature enables an SSH client to make a secure, encrypted connection to a Cisco device, and the SSH client feature enables a Cisco device to make a secure, encrypted connection to another Cisco device or to any other device running an SSH server. IPv6 enhancements to SSH consist of support for IPv6 addresses that enable a Cisco device to accept and establish secure, encrypted connections with remote IPv6 nodes over an IPv6 transport.

SNMP over an IPv6 Transport

Simple Network Management Protocol (SNMP) can be configured over IPv6 transport so that an IPv6 host can perform SNMP queries and receive SNMP notifications from a device running IPv6 software. The SNMP agent and related MIBs have been enhanced to support IPv6 addressing. This feature uses the data encryption standard (3DES) and advanced encryption standard (AES) message encryption.

- Cisco IPv6 MIBs, page 416
- MIBs Supported for IPv6, page 417

Cisco IPv6 MIBs

Cisco has long supported IP-MIB and IP-FORWARD-MIB in IPv4. CISCO-IETF-IP-MIB and CISCO-IETF-IP-FORWARDING-MIB are IPv6 MIBs that are defined as being protocol-independent, but are

implemented only for IPv6 objects and tables. IP-MIB and IP-FORWARD-MIB adhere to RFC 4293 and RFC 4292 standards, as follows:

- The upgrade is backward-compatible; all IP-MIB and IP-FORWARD-MIB objects and tables still appear.
- IP-MIB and IP-FORWARD-MIB include definitions of new IPv6-only, IPv4-only, and protocol-version independent (PVI) objects and tables.

CISCO-IETF-IP-MIB and CISCO-IETF-IP-FORWARDING-MIB were removed from the Cisco releases in which CISCO-IETF-IP-MIB and CISCO-IETF-IP-FORWARDING-MIB were applied. Information in CISCO-IETF-IP-MIB and CISCO-IETF-IP-FORWARDING-MIB is included IP-MIB and IP-FORWARD-MIB.

MIBs Supported for IPv6

The following MIBs are supported for IPv6:

- CISCO-CONFIG-COPY-MIB
- CISCO-CONFIG-MAN-MIB
- CISCO-DATA-COLLECTION-MIB
- CISCO-FLASH-MIB
- CISCO-SNMP-TARGET-EXT-MIB
- ENTITY-MIB
- IP-FORWARD-MIB
- IP-MIB
- NOTIFICATION-LOG-MIB
- SNMP-TARGET-MIB

CISCO-CONFIG-COPY-MIB and CISCO-FLASH-MIB support IPv6 addressing when TFTP, remote copy protocol (rcp), or FTP is used.

Cisco IPv6 Embedded Management Components

Cisco embedded management components have IPv6-compliant operability in IPv6 and dual-stack IPv6 and IPv4 networks.

Syslog

The Cisco system message logging (syslog) process in IPv6 allows users to log syslog messages to external syslog servers and hosts with IPv6 addresses. This implementation allows user to specify an IPv4-based logging host (syslog server) by providing the host's IP address in IPv4 format (for example, 192.168.0.0) or IPv6 format (for example, 2001:DB8:A00:1::1/64).

CNS Agents

IPv6 addressing is supported in the Cisco Networking Services (CNS) subsystem. CNS is a foundation technology for linking users to networking services, and it provides the infrastructure for the automated configuration of large numbers of network devices. Many IPv6 networks are complex, with many devices, and each device must be configured individually. When standard configurations do not exist or have been modified, the time involved in initial installation and subsequent upgrading is considerable. ISPs need a method for sending out partial configurations to introduce new services.

To address all these issues, CNS was designed to provide "plug-and-play" network services using a central directory service and distributed agents. CNS features include CNS agents and a flow-through provisioning structure. CNS flow-through provisioning uses the CNS configuration and event agents to provide an automated workflow, eliminating the need for an onsite technician.

IPv6 addressing supports the CNS agents described in the following sections:

- CNS Configuration Agent, page 418
- CNS Event Agent, page 418
- CNS EXEC Agent, page 418
- CNS Image Agent, page 418

CNS Configuration Agent

The CNS configuration agent is involved in the initial configuration and subsequent partial configurations on a Cisco device. The configuration agent uses a CNS configuration engine to provide methods for automating initial Cisco device configurations, incremental configurations, and synchronized configuration updates, and the configuration engine reports the status of the configuration load as an event to which a network monitoring or workflow application can subscribe.

CNS Event Agent

The CNS event agent provides a transport connection to the CNS event bus for all other CNS agents. No event can be sent to the device by the configuration engine until the CNS event agent is operational and has successfully built a connection between the configuration engine and the device.

The event agent uses a CNS configuration engine to provide methods for automating initial Cisco device configurations, incremental configurations, and synchronized configuration updates.

CNS EXEC Agent

The CNS EXEC agent allows a remote application to execute a CLI command in EXEC mode on a Cisco device by sending an event message that contains the command.

CNS Image Agent

Administrators maintaining large networks of Cisco devices need an automated mechanism to load image files onto large numbers of remote devices. Network management applications are useful to determine which images to run and how to manage images received from the Cisco online software center. Other image distribution solutions do not scale to cover thousands of devices and cannot distribute images to devices behind a firewall or using Network Address Translation (NAT). The CNS image agent enables the managed device to initiate a network connection and request an image download allowing devices using NAT, or behind firewalls, to access the image server.

The CNS image agent can be configured to use the CNS event bus. To use the CNS event bus, the CNS event agent must be enabled and connected to the CNS event gateway in the CNS Configuration Engine.

The CNS image agent can also use an HTTP server that understands the CNS image agent protocol. Deployment of CNS image agent operations can use both the CNS event bus and an HTTP server.

Config Logger

Config logger tracks and reports configuration changes. Config logger supports two content types:

- Plain text--With plain-text format, the config logger reports configuration changes only.
- XML--The config logger uses XML to report the configuration change details (for example, what changed, who changed it, when changes were made, parser return code [PRC] values, and incremental NVGEN results).

HTTP(S) IPv6 Support

This feature allows the HTTP(S) client and server to support IPv6 addresses.

The HTTP server in Cisco software can service requests from both IPv6 and IPv4 HTTP clients. When the HTTP(S) server accepts a connection from a client, the server determines whether the client is an IPv4 or IPv6 host. The address family, IPv4 or IPv6, for the accept socket call is then chosen accordingly. The listening socket continues to listen for both IPv4 and IPv6 connections.

The HTTP client in Cisco software can send requests to both IPv4 and IPv6 HTTP servers.

When you use the IPv6 HTTP client, URLs with literal IPv6 addresses must be formatted using the rules listed in RFC 2732.

TCL

Tool command language (TCL) is used in Cisco software for IPv6 to support features such as embedded syslog manager (ESM), embedded event manager (EEM), interactive voice response (IVR), and tclsh parser mode. TCL supports both initiating (client) and listening (server) sockets.

NETCONF

The Network Configuration Protocol (NETCONF) defines a mechanism through which a network device can be managed, configuration data information can be retrieved, and new configuration data can be uploaded and manipulated. NETCONF uses XML-based data encoding for the configuration data and protocol messages.

SOAP Message Format

Using the Service-Oriented Access Protocol (SOAP) provides a way to format the layout of Cisco Networking Services (CNS) messages in a consistent manner. SOAP is intended for exchanging structured information in a decentralized, distributed environment. SOAP uses XML technologies to define an extensible messaging framework that provides a message format that can be exchanged over a variety of underlying protocols.

Within the SOAP message structure, there is a security header that enables CNS notification messages to authenticate user credentials.

IP SLAs for IPv6

Cisco IP Service Level Agreements (SLAs) are a portfolio of technology embedded in most devices that run Cisco software that allows Cisco customers to analyze IPv6 service levels for IPv6 applications and

services, increase productivity, lower operational costs, and reduce the frequency of network outages. IP SLAs uses active traffic monitoring--the generation of traffic in a continuous, reliable, and predictable manner--for measuring network performance.

The following Cisco IP SLAs are supported for IPv6:

- Internet Control Message Protocol (ICMP) echo operation--Used to monitor end-to-end response time between a Cisco device and other devices using IPv4 or IPv6. ICMP echo is useful for troubleshooting network connectivity issues.
- TCP connect operation--Used to measure the response time taken to perform a TCP Connect operation between a Cisco device and other devices using IPv4 or IPv6.
- User Datagram Protocol (UDP) echo operation--Used to monitor end-to-end response time between a Cisco router and devices using IPv4 or IPv6 .
- UDP jitter operation--Used to analyze round-trip delay, one-way delay, one-way jitter, one-way packet loss, and connectivity in networks that carry UDP traffic in IPv4 or IPv6 networks.
- UDP jitter operation--Used to monitor VoIP quality levels in your network, allowing you to guarantee VoIP quality levels to your users in IPv4 or IPv6 networks.

How to Implement IPv6 for Network Management

Enabling Telnet Access to an IPv6 Device and Establishing a Telnet Session

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 host** *name* [*port*] *ipv6-address*
4. **line** [**aux** | **console** | **tty** | **vty**] *line-number* [*ending-line-number*]
5. **password** *password*
6. **login** [**local** | **tacacs**]
7. **ipv6 access-class** *ipv6-access-list-name* {**in** | **out**]
8. **telnet** *host* [*port*] [*keyword*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 host** *name* [*port*] *ipv6-address* **Example:** `Device(config)# ipv6 host cisco-sj 2001:DB8:20:1::12`	Defines a static hostname-to-address mapping in the hostname cache.
Step 4	**line** [**aux** | **console** | **tty** | **vty**] *line-number* [*ending-line-number*] **Example:** `Device(config)# line vty 0 4`	Creates a vty interface.
Step 5	**password** *password* **Example:** `Device(config)# password hostword`	Creates a password that enables Telnet.
Step 6	**login** [**local** | **tacacs**] **Example:** `Device(config)# login tacacs`	(Optional) Enables password checking at login.
Step 7	**ipv6 access-class** *ipv6-access-list-name* {**in** | **out**] **Example:** `Device(config)# ipv6 access-list hostlist`	(Optional) Adds an IPv6 access list to the line interface. • Using this command restricts remote access to sessions that match the access list.

	Command or Action	Purpose
Step 8	**telnet** *host* [*port*] [*keyword*] **Example:** `Device(config)# telnet cisco-sj`	Establishes a Telnet session from a device to a remote host using either the hostname or the IPv6 address. • The Telnet session can be established to a device name or to an IPv6 address.

Enabling SSH on an IPv6 Device

This task is optional. If you do not configure SSH parameters, then the default values will be used.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip ssh** [**timeout** *seconds* | **authentication-retries** *integer*]
4. **exit**
5. **ssh** [**-v** { **1** | **2** } | **c** { **3des** | **aes128-cbc** | **aes192-cbc** | **aes256-cbc** } | **-l** *userid* | **-l** *userid:vrfname number ip-address ip-address* | **-l** *userid:rotary number ip-address* | **-m** { **hmac-md5** | **hmac-md5-96** | **hmac-sha1** | **hmac-sha1-96** } | **-o numberofpasswordprompts** *n* | **-p** *port-num*] { **ip-addr** | **hostname**} [**command** | **-vrf**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip ssh** [**timeout** *seconds*	**authentication-retries** *integer*] **Example:** `Device(config)# IP ssh timeout 100 authentication-retries 2`	Configures SSH control variables on your device.

	Command or Action	Purpose
Step 4	**exit** **Example:** `Device(config)# exit`	Exits configuration mode, and returns the device to privileged EXEC mode.
Step 5	**ssh** [**-v** { **1** | **2** } | **c** { **3des** | **aes128-cbc** | **aes192-cbc** | **aes256-cbc** } | **-l** *userid* | **-l** *userid:vrfname number ip-address ip-address* | **-l** *userid:rotary number ip-address* | **-m** { **hmac-md5** | **hmac-md5-96** | **hmac-sha1** | **hmac-sha1-96** } | **-o numberofpasswordprompts** *n* | **-p** *port-num*] { **ip-addr** | **hostname**} [**command** | **-vrf**] **Example:** `Device# ssh -l userid1 2001:db8:2222:1044::72`	Starts an encrypted session with a remote networking device.

Configuring an SNMP Notification Server over IPv6

Use an SNMP community string to define the relationship between the SNMP manager and the agent. The community string acts like a password to regulate access to the agent on the device. Optionally, you can specify one or more of the following characteristics associated with the string:

- An access list of IP addresses of the SNMP managers that are permitted to use the community string to gain access to the agent.
- A MIB view, which defines the subset of all MIB objects accessible to the given community.
- Read and write or read-only permission for the MIB objects accessible to the community.

You can configure one or more community strings. To remove a specific community string, use the **no snmp-server community** command.

The **snmp-server host** command specifies which hosts will receive SNMP notifications, and whether you want the notifications sent as traps or inform requests. The **snmp-server enable traps** command globally enables the production mechanism for the specified notification types (such as Border Gateway Protocol [BGP] traps, config traps, entity traps, and Hot Standby Router Protocol [HSRP] traps).

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server community** *string* [**view** *view-name*] [**ro** | **rw**] [**ipv6** *nacl*] [*access-list-number*]
4. **snmp-server engineID remote** {*ipv4-ip-address* | *ipv6-address*} [**udp-port** *udp-port-number*] [**vrf** *vrf-name*] *engineid-string*
5. **snmp-server group** *group-name* {**v1** | **v2c** | **v3** {**auth** | **noauth** | **priv**}} [**context** *context-name*] [**read** *read-view*] [**write** *write-view*] [**notify** *notify-view*] [**access** [**ipv6** *named-access-list*] {*acl-number* | *acl-name*}]
6. **snmp-server host** {*hostname* | *ip-address*} [**vrf** *vrf-name*] [**traps** | **informs**] [**version** {**1** | **2c** | **3** [**auth** | **noauth** | **priv**]}] *community-string* [**udp-port** *port*] [*notification-type*]
7. **snmp-server user** *username group-name* [**remote** *host* [**udp-port** *port*]] {**v1** | **v2c** | **v3** [**encrypted**] [**auth** {**md5** | **sha**} *auth-password*]} [**access** [**ipv6** *nacl*] [**priv** {**des** | **3des** | **aes** {**128** | **192** | **256**}} *privpassword*] {*acl-number* | *acl-name*}]
8. **snmp-server enable traps** [*notification-type*] [**vrrp**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server community** *string* [**view** *view-name*] [**ro**	**rw**] [**ipv6** *nacl*] [*access-list-number*] **Example:** `Device(config)# snmp-server community mgr view restricted rw ipv6 mgr2`	Defines the community access string.
Step 4	**snmp-server engineID remote** {*ipv4-ip-address*	*ipv6-address*} [**udp-port** *udp-port-number*] [**vrf** *vrf-name*] *engineid-string* **Example:** `Device(config)# snmp-server engineID remote 3ffe:b00:c18:1::3/127 remotev6`	(Optional) Specifies the name of the remote SNMP engine (or copy of SNMP).

	Command or Action	Purpose
Step 5	**snmp-server group** *group-name* {**v1** | **v2c** | **v3** {**auth** | **noauth** | **priv**}} [**context** *context-name*] [**read** *read-view*] [**write** *write-view*] [**notify** *notify-view*] [**access** [**ipv6** *named-access-list*] {*acl-number* | *acl-name*}] **Example:** `Device(config)# snmp-server group public v2c access ipv6 public2`	(Optional) Configures a new SNMP group, or a table that maps SNMP users to SNMP views.
Step 6	**snmp-server host** {*hostname* | *ip-address*} [**vrf** *vrf-name*] [**traps** | **informs**] [**version** {**1** | **2c** | **3** [**auth** | **noauth** | **priv**]}] *community-string* [**udp-port** *port*] [*notification-type*] **Example:** `Device(config)# snmp-server host host1.com 2c vrf trap-vrf`	Specifies the recipient of an SNMP notification operation. • Specifies whether you want the SNMP notifications sent as traps or informs, the version of SNMP to use, the security level of the notifications (for SNMPv3), and the recipient (host) of the notifications.
Step 7	**snmp-server user** *username group-name* [**remote** *host* [**udp-port** *port*]] {**v1** | **v2c** | **v3** [**encrypted**] [**auth** {**md5** | **sha**} *auth-password*]} [**access** [**ipv6** *nacl*] [**priv** {**des** | **3des** | **aes** {**128** | **192** | **256**}} *privpassword*] {*acl-number* | *acl-name*}] **Example:** `Device(config)# snmp-server user user1 bldg1 remote 3ffe:b00:c18:1::3/127 v2c access ipv6 public2`	(Optional) Configures a new user to an existing SNMP group. **Note** You cannot configure a remote user for an address without first configuring the engine ID for that remote host. This is a restriction imposed in the design of these commands; if you try to configure the user before the host, you will receive a warning message, and the command will not be executed.
Step 8	**snmp-server enable traps** [*notification-type*] [**vrrp**] **Example:** `Device(config)# snmp-server enable traps bgp`	Enables sending of traps or informs, and specifies the type of notifications to be sent. • If a value for the *notification-type* argument is not specified, all supported notification will be enabled on the device. • To discover which notifications are available on your device, enter the **snmp-server enable traps ?** command.

Configuring Cisco IPv6 Embedded Management Components

Most IPv6 embedded management components are enabled automatically when IPv6 is enabled and do not need further configuration. However, you may want to perform the following task to disable HTTP access to a device:

- Configuring Syslog over IPv6, page 426
- Disabling HTTP Access to an IPv6 Device, page 426

Configuring Syslog over IPv6

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **logging host** {{*ip-address* | *hostname*} | {**ipv6** *ipv6-address* | *hostname*}} [**transport** {**udp** [**port** *port-number*] | **tcp** [**port** *port-number*] [**audit**]}] [**xml** | **filtered** [**stream** *stream-id*]] [**alarm** [*severity*]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**logging host** {{*ip-address* | *hostname*} | {**ipv6** *ipv6-address* | *hostname*}} [**transport** {**udp** [**port** *port-number*] | **tcp** [**port** *port-number*] [**audit**]}] [**xml** | **filtered** [**stream** *stream-id*]] [**alarm** [*severity*]] **Example:** `Device(config)# logging host ipv6 AAAA:BBBB:CCCC:DDDD::FFFF`	Logs system messages and debug output to a remote host.

Disabling HTTP Access to an IPv6 Device

HTTP access over IPv6 is automatically enabled if an HTTP server is enabled and the device has an IPv6 address. If the HTTP server is not required, it should be disabled.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no ip http server**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no ip http server** **Example:** `Device(config)# no ip http server`	Disables HTTP access.

Configuration Examples for Implementing IPv6 for Network Management

Examples: Enabling Telnet Access to an IPv6 Device

The following examples provide information on how to enable Telnet and start a session to or from an IPv6 device. In the following example, the IPv6 address is specified as 2001:DB8:20:1::12, and the hostname is specified as cisco-sj. The **show host** command is used to verify this information.

```
Device# configure terminal
Device(config)# ipv6 host cisco-sj 2001:DB8:20:1::12
Device(config)# end
Device# show host
Default domain is not set
Name/address lookup uses static mappings
Codes:UN - unknown, EX - expired, OK - OK, ?? - revalidate
       temp - temporary, perm - permanent
       NA - Not Applicable None - Not defined
Host                 Port  Flags      Age Type   Address(es)
cisco-sj             None  (perm, OK)  0  IPv6 2001:DB8:20:1::12
```

To enable Telnet access to a device, create a vty interface and password:

```
Device(config)# line vty 0 4
```

```
password lab
login
```

To use Telnet to access the device, you must enter the password:

```
Device# telnet cisco-sj
Trying cisco-sj (2001:DB8:20:1::12)... Open
User Access Verification
Password:
cisco-sj
.
.
.
verification
```

It is not necessary to use the **telnet** command. Specifying either the hostname or the address is sufficient, as shown in the following examples:

```
Device# cisco-sj
```

or

```
Device# 2001:DB8:20:1::12
```

To display the IPv6 connected user (line 130) on the device to which you are connected, use the **show users** command:

```
Device# show users
    Line       User       Host(s)              Idle       Location
*  0 con 0                idle                 00:00:00
 130 vty 0                idle                 00:00:22   8800::3
```

Note that the address displayed is the IPv6 address of the source of the connection. If the hostname of the source is known (either through a domain name server [DNS] or locally in the host cache), then it is displayed instead:

```
Device# show users
    Line       User       Host(s)              Idle       Location
*  0 con 0                idle                 00:00:00
 130 vty 0                idle                 00:02:47   cisco-sj
```

If the user at the connecting device suspends the session with ^6x and then enters the **show sessions** command, the IPv6 connection is displayed:

```
Device# show sessions
Conn Host               Address             Byte  Idle Conn Name
*  1 cisco-sj 2001:DB8:20:1::12      0     0 cisco-sj
```

The Conn Name field shows the hostname of the destination only if it is known. If it is not known, the output might look similar to the following:

```
Device# show sessions
Conn Host               Address             Byte  Idle Conn Name
*  1 2001:DB8:20:1::12 2001:DB8:20:1::12      0     0 2001:DB8:20:1::12
```

Example: Disabling HTTP Access to the Device

In the following example, the **show running-config** command is used to show that HTTP access is disabled on the device:

```
Device# show running-config

Building configuration...
```

```
!
Current configuration : 1490 bytes
!
version 12.2
!
hostname Device
!
no ip http server
!
line con 0
line aux 0
line vty 0 4
```

Examples: Configuring an SNMP Notification Server over IPv6

The following example permits any SNMP to access all objects with read-only permission using the community string named public. The device also will send Border Gateway Protocol (BGP) traps to the IPv4 host 172.16.1.111 and IPv6 host 3ffe:b00:c18:1::3/127 using SNMPv1 and to the host 172.16.1.27 using SNMPv2c. The community string named public will be sent with the traps.

```
Device(config)# snmp-server community public
Device(config)# snmp-server enable traps bgp
Device(config)# snmp-server host 172.16.1.27 version 2c public
Device(config)# snmp-server host 172.16.1.111 version 1 public
Device(config)# snmp-server host 3ffe:b00:c18:1::3/127 public
```

Example: Associate an SNMP Server Group with Specified Views

In the following example, the SNMP context A is associated with the views in SNMPv2c group GROUP1 and the IPv6 named access list public2:

```
Device(config)# snmp-server context A
Device(config)# snmp mib community-map commA context A target-list commAVpn
Device(config)# snmp mib target list commAVpn vrf CustomerA
Device(config)# snmp-server view viewA ciscoPingMIB included
Device(config)# snmp-server view viewA ipForward included
Device(config)# snmp-server group GROUP1 v2c context A read viewA write viewA notify
access ipv6 public2
```

Example: Create an SNMP Notification Server

The following example configures the IPv6 host as the notification server:

```
Device> enable
Device# configure terminal
Device(config)# snmp-server community mgr view restricted rw ipv6 mgr2
Device(config)# snmp-server engineID remote 3ffe:b00:c18:1::3/127 remotev6
Device(config)# snmp-server group public v2c access ipv6 public2
Device(config)# snmp-server host host1.com 2c vrf trap-vrf
Device(config)# snmp-server user user1 bldg1 remote 3ffe:b00:c18:1::3/127 v2c access ipv6
public2
Device(config)# snmp-server enable traps bgp
Device(config)# exit
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 supported features	"Start Here: Cisco IOS Software Release Specifics for IPv6 Features ," *Cisco IOS IPv6 Configuration Guide*
Basic IPv6 configuration tasks	"Implementing IPv6 Addressing and Basic Connectivity ," *Cisco IOS IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*
SSH configuration information	*Cisco IOS Security Command Reference*
IPv4 CNS, SOAP	"Cisco Networking Services," *Cisco IOS Network Management Configuration Guide*
NETCONF	"Network Configuration Protocol," *Cisco IOS Network Management Configuration Guide*
IP SLAs for IPv6	• IP SLAs--Analyzing IP Service Levels Using the ICMP Echo Operation • IP SLAs--Analyzing IP Service Levels Using the TCP Connect Operation • IP SLAs--Analyzing IP Service Levels Using the UDP Echo Operation • IP SLAs--Analyzing IP Service Levels Using the UDP Jitter Operation • IP SLAs--Analyzing VoIP Service Levels Using the UDP Jitter Operation

Standards

Standard	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIB	MIBs Link
• CISCO-CONFIG-COPY-MIB • CISCO-CONFIG-MAN-MIB • CISCO-DATA-COLLECTION-MIB • CISCO-FLASH-MIB • CISCO-RTTMON-IPv6-MIB • CISCO-SNMP-TARGET-EXT-MIB • ENTITY-MIB • IP-FORWARD-MIB • IP-MIB • NOTIFICATION-LOG-MIB • SNMP-TARGET-MIB	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2732	*Format for Literal IPv6 Addresses in URLs*
RFC 3414	*User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)*
RFC 3484	*Default Address Selection for Internet Protocol version 6 (IPv6)*
RFC 4292	IP Forwarding Table MIB
RFC 4293	*Management Information Base for the Internet Protocol (IP)*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing IPv6 for Network Management

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 20 Feature Information for Managing Cisco IOS Applications over IPv6

Feature Name	Releases	Feature Information
CNS Agents for IPv6	12.2(33)SB 12.2(33)SRC 12.2(50)SY 12.4(20)T	CNS configuration and event agents use a CNS configuration engine to provide methods for automating initial Cisco IOS device configurations, incremental configurations, and synchronized configuration updates, and the configuration engine reports the status of the configuration load as an event to which a network monitoring or workflow application can subscribe.
HTTP(S) IPv6 Support	12.2(33)SB 12.2(33)SRC 12.2(50)SY 12.4(20)T	This feature enhances the HTTP(S) client and server to support IPv6 addresses.
IP SLAs for IPv6	12.2(33)SB 12.2(33)SRC 12.2(50)SY 12.4(20)T 15.0(1)S	IP SLAs are supported for IPv6.
IPv6 for Config Logger	12.2(33)SB 12.2(33)SRC 12.2(50)SY 12.4(20)T	Config logger tracks and reports configuration changes.
IPv6 NETCONF Support	12.2(33)SB 12.2(33)SRC 12.2(50)SY 12.4(20)T	The Network Configuration Protocol (NETCONF) defines a simple mechanism through which a network device can be managed, configuration data information can be retrieved, and new configuration data can be uploaded and manipulated.

Feature Name	Releases	Feature Information
IPv6--syslog over IPv6	12.2(33)SB 12.2(33)SRC 12.2(33)SXI 12.4(4)T	The Cisco IOS syslog process in IPv6 allows users to log syslog messages to external syslog servers and hosts with IPv6 addresses.
IPv6 Services--IP-FORWARD-MIB Support	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(15)T 12.3 12.3(2)T 12.4 12.4(2)T	A MIB is a database of objects that can be managed on a device. The managed objects, or variables, can be set or read to provide information on the network devices and interfaces.
IPv6 Services--IP-MIB Support	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(15)T 12.3 12.3(2)T 12.4 12.4(2)T	A MIB is a database of objects that can be managed on a device. The managed objects, or variables, can be set or read to provide information on the network devices and interfaces.
IPv6 Services--RFC 4293 IP-MIB (IPv6 only) and RFC 4292 IP-FORWARD-MIB (IPv6 only)	12.2(33)SB 12.2(58)SE 12.2(54)SG 12.2(33)SRC 12.2(50)SY 15.1(3)T	IP-FORWARD-MIB and IP-MIB were updated to RFC 4292 and RFC 4293 standards, respectively.
IPv6 Support for TCL	12.2(33)SRC 12.2(50)SY 12.4(20)T	IPv6 supports TCL.
IPv6 Support in SOAP	12.2(33)SB 12.2(33)SRC 12.2(50)SY 12.4(20)T	SOAP is a protocol intended for exchanging structured information in a decentralized, distributed environment.
SNMP over IPv6	12.0(27)S 12.2(33)SRB 12.2(33)SXI 12.3(14)T 12.4 12.4(2)T	SNMP can be configured over IPv6 transport so that an IPv6 host can perform SNMP queries and receive SNMP notifications from a device running Cisco IOS IPv6.
SNMPv3 - 3DES and AES Encryption Support	12.2(33)SB 12.2(33)SRB 12.2(33)SXI 12.4(2)T	SNMP for IPv6 supports 3DES and AES encryption.

Feature Name	Releases	Feature Information
SSH over an IPv6 Transport	12.0(22)S 12.2(8)T 12.2(14)S 12.2(28)SB 12.2(33)SRA 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	SSH in IPv6 functions the same and offers the same benefits as SSH in IPv4--the SSH Server feature enables an SSH client to make a secure, encrypted connection to a Cisco router and the SSH Client feature enables a Cisco router to make a secure, encrypted connection to another Cisco router or to any other device running an SSH server.
Telnet Access over IPv6	12.0(22)S 12.2(2)T 12.2(14)S 12.2(28)SB 12.3 12.3(2)T 12.4 12.4(2)T	The Telnet client and server in the Cisco IOS software support IPv6 connections. A user can establish a Telnet session directly to the router using an IPv6 Telnet client, or an IPv6 Telnet connection can be initiated from the router.
TFTP File Downloading for IPv6	12.0(22)S 12.2(2)T 12.2(14)S 12.2(28)SB 12.3 12.3(2)T 12.4 12.4(2)T	IPv6 supports TFTP file downloading and uploading.

Implementing Mobile IPv6

Mobile IP is part of both IPv4 and IPv6 standards. Mobile IP allows a host device to be identified by a single IP address even though the device may move its physical point of attachment from one network to another. Regardless of movement between different networks, connectivity at the different points is achieved seamlessly without user intervention. Roaming from a wired network to a wireless or wide-area network is also done with ease. Mobile IP provides ubiquitous connectivity for users, whether they are within their enterprise networks or away from home.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Implementing Mobile IPv6

When using the network mobility (NEMO) basic support protocol feature, users should not enable any IPv6 routing protocols on any of the roaming interfaces.

Information About Implementing Mobile IPv6

Mobile IPv6 Overview

Mobile IPv4 provides an IPv4 node with the ability to retain the same IPv4 address and maintain uninterrupted network and application connectivity while traveling across networks. In Mobile IPv6, the IPv6 address space enables Mobile IP deployment in any kind of large environment. No foreign agent is needed to use Mobile IPv6.

System infrastructures do not need an upgrade to accept Mobile IPv6 nodes. IPv6 autoconfiguration simplifies mobile node (MN) Care of Address (CoA) assignment.

Mobile IPv6 benefits from the IPv6 protocol itself; for example, Mobile IPv6 uses IPv6 option headers (routing, destination, and mobility) and benefits from the use of neighbor discovery.

Mobile IPv6 provides optimized routing, which helps avoid triangular routing. Mobile IPv6 nodes work transparently even with nodes that do not support mobility (although these nodes do not have route optimization).

Mobile IPv6 is fully backward-compatible with existing IPv6 specifications. Therefore, any existing host that does not understand the new mobile messages will send an error message, and communications with the mobile node will be able to continue, albeit without the direct routing optimization.

How Mobile IPv6 Works

To implement Mobile IPv6, you need a home agent on the home subnet on which the mobile node's home address resides. The IPv6 home address (HA) is assigned to the mobile node. The mobile node obtains a new IPv6 address (the CoA) on networks to which it connects. The home agent accepts BUs from the mobile node informing the agent of the mobile node's location. The home agent then acts as proxy for the mobile node, intercepting traffic to the mobile node's home address and tunneling it to the mobile node.

The mobile node informs a home agent on its original home network about its new address, and the correspondent node communicates with the mobile node about the CoA. Because of the use of ingress filtering, the mobile node reverses tunnel return traffic to the home agent, so that the mobile node source address (that is, its home address) will always be topographically correct.

Mobile IPv6 is the ability of a mobile node to bypass the home agent when sending IP packets to a correspondent node. Optional extensions make direct routing possible in Mobile IPv6, though the extensions might not be implemented in all deployments of Mobile IPv6.

Direct routing is built into Mobile IPv6, and the direct routing function uses the IPv6 routing header and the IPv6 destination options header. The routing header is used for sending packets to the mobile node using its current CoA, and the new home address destination option is used to include the mobile node's home address, because the current CoA is the source address of the packet.

IPv6 NEMO

The NEMO basic support protocol enables mobile IPv6 networks to attach to different points in the Internet. This protocol is an extension of Mobile IPv6 and allows session continuity for every node in the mobile network as the network moves. NEMO also allows every node in the mobile network to be reachable while the user is moving. The mobile router, which connects the network to the Internet, runs the

NEMO basic support protocol with its home agent (HA). NEMO allows network mobility to be transparent to the nodes inside the mobile network.

The NEMO router maintains a mobile route, which is the default route for IPv6 over the roaming interface.

Mobile IPv6 Home Agent

The home agent is one of three key components in Mobile IPv6. The home agent works with the correspondent node and mobile node to enable Mobile IPv6 functionality:

- Home agent--The home agent maintains an association between the mobile mode's home IPv4 or IPv6 address and its CoA (loaned address) on the foreign network.
- Correspondent node--The correspondent node is the destination IPv4 or IPv6 host in session with a mobile node.
- Mobile node--An IPv4 or IPv6 host that maintains network connectivity using its home IPv4 or IPv6 address, regardless of the link (or network) to which it is connected.

Binding Cache in Mobile IPv6 Home Agent

A separate binding cache is maintained by each IPv6 node for each of its IPv6 addresses. When the router sends a packet, it searches the binding cache for an IPv6 address before it searches the neighbor discovery conceptual destination cache.

The binding cache for any one of a node's IPv6 addresses may contain one entry for each mobile node home address. The contents of all of a node's binding cache entries are cleared when it reboots.

Binding cache entries are marked either as home registration or correspondent registration entries. A home registration entry is deleted when its binding lifetime expires; other entries may be replaced at any time through a local cache replacement policy.

Binding Update List in Mobile IPv6 Home Agent

A binding update (BU) list is maintained by each mobile node. The BU list records information for each BU sent by this mobile node whose lifetime has not yet expired. The BU list includes all BUs sent by the mobile node--those bindings sent to correspondent nodes, and those bindings sent to the mobile node's home agent.

The mobility extension header has a new routing header type and a new destination option, and it is used during the BU process. This header is used by mobile nodes, correspondent nodes, and home agents in all messaging related to the creation and management of bindings.

Home Agents List

A home agents list is maintained by each home agent and each mobile node. The home agents list records information about each home agent from which this node has recently received a router advertisement in which the home agent (H) bit is set.

Each home agent maintains a separate home agents list for each link on which it is serving as a home agent. This list is used by a home agent in the dynamic home agent address discovery mechanism. Each roaming

mobile node also maintains a home agents list that enables it to notify a home agent on its previous link when it moves to a new link.

NEMO-Compliant Home Agent

Protocol extensions to Mobile IPv6 are used to enable support for network mobility. The extensions are backward-compatible with existing Mobile IPv6 functionality. A NEMO-compliant home agent can operate as a Mobile IPv6 home agent.

The dynamic home agent address discovery (DHAAD) mechanism allows a mobile node to discover the address of the home agent on its home link. The following list describes DHAAD functionality and features:

- The mobile router sends Internet Control Message Protocol (ICMP) home agent address discovery requests to the Mobile IPv6 home agent's anycast address for the home subnet prefix.
- A new flag (R) is introduced in the DHAAD request message, indicating the desire to discover home agents that support mobile routers. This flag is added to the DHAAD reply message as well.
- On receiving the home agent address discovery reply message, the mobile router discovers the home agents operating on the home link.
- The mobile router attempts home registration to each of the home agents until its registration is accepted. The mobile router waits for the recommended length of time between its home registration attempts with each of its home registration attempts.

- Implicit Prefix Registration, page 438
- Explicit Prefix Registration, page 438

Implicit Prefix Registration

When using implicit prefix registration, the mobile router does not register any prefixes as part of the binding update with its home agent. This function requires a static configuration at the home agent, and the home agent must have the information of the associated prefixes with the given mobile router for it to set up route forwarding.

Explicit Prefix Registration

When using explicit prefix registration, the mobile router presents a list of prefixes to the home agent as part of the binding update procedure. If the home agent determines that the mobile router is authorized to use these prefixes, it sends a bind acknowledgment message.

Packet Headers in Mobile IPv6

The basic IPv6 packet header has 8 fields with a total size of 40 octets (320 bits). Fields were removed from the IPv6 header compared with the IPv4 header because, in IPv6, fragmentation is not handled by routers and checksums at the network layer are not used. Instead, fragmentation in IPv6 is handled by the source of a packet and checksums at the data link layer and transport layer are used. Additionally, the basic IPv6 packet header and options field are aligned to 64 bits, which can facilitate the processing of IPv6 packets.

Mobile IPv6 uses the routing and destination option headers for communications between the mobile node and the correspondent node. The new mobility option header is used only for the BU process.

Several ICMP message types have been defined to support Mobile IPv6. IPv6 access lists can be configured to allow IPv6 access list entries matching Mobile-IPv6-specific ICMP messages to be

configured and to allow the definition of entries to match packets containing Mobile IPv6 extension headers.

IPv6 Neighbor Discovery with Mobile IPv6

The IPv6 neighbor discovery feature has the following modifications to allow the feature to work with Mobile IPv6:

- Modified router advertisement message format--has a single flag bit that indicates home agent service
- Modified prefix information option format--allows a router to advertise its global address
- New advertisement interval option format
- New home agent information option format
- Changes to sending router advertisements
- Provide timely movement detection for mobile nodes

- IPv6 Neighbor Discovery Duplicate Address Detection in NEMO, page 439

IPv6 Neighbor Discovery Duplicate Address Detection in NEMO

IPv6 routers are required to run duplicate address detection (DAD) on all IPv6 addresses obtained in stateless and stateful autoconfiguration modes before assigning them to any of its interfaces. Whenever an mobile router roams and obtains an IPv6 address, the mobile router must perform DAD on the newly obtained care-of address and on its link-local address in order to avoid address collisions.

However, the DAD feature adds significant handoff delays in certain Layer 2 environments. These delays may be avoided by using optimistic DAD techniques. NEMO supports optimization options for omitting DAD on care-of address or on both the care-of address and link-local address.

Mobile IPv6 Tunnel Optimization

Mobile IPv6 tunnel optimization enables routing over a native IPv6 tunnel infrastructure, allowing Mobile IPv6 to use all IPv6 tunneling infrastructure features, such as Cisco Express Forwarding switching support.

After the home agent receives a valid BU request from a mobile node, it sets up its endpoint of the bidirectional tunnel. This process involves creating a logical interface with the encapsulation mode set to IPv6/IPv6, the tunnel source to the home agent’s address on the mobile node’s home link, and the tunnel destination set to the mobile node’s registered care-of address. A route will be inserted into the routing table for the mobile node’s home address via the tunnel.

IPv6 Host Group Configuration

Users can create mobile user or group policies using the IPv6 host group configuration. The host group profile lookup interface will allow the lookup of the profile associated with the sender of the BU using any of the search keys:

- Profile name
- IPv6 address
- Network address identifier (NAI)

The host profile lookup interface also specifies the authentication properties for the IPv6 mobile node by creating either a unidirectional or bidirectional security parameter index (SPI).

A group profile is activated after the SPI option is configured and either an NAI or an IPv6 address is configured. In addition, a profile is deactivated if the minimum required options are not configured. If any

active profile that has active bindings gets deactivated or removed, all bindings associated to that profile are revoked.

- Mobile IPv6 Node Identification Based on NAI, page 440
- Authentication Protocol for Mobile IPv6, page 440

Mobile IPv6 Node Identification Based on NAI

A mobile node can identify itself using its home address as an identifier. The Mobile IPv6 protocol messages use this identifier in their registration messages. However, for certain deployments it is essential that the mobile node has the capability to identify itself using a logical identifier, such as NAI, rather than a network address. The mobile node identifier option for Mobile IPv6 allows a mobile node to be identified by NAI rather than IPv6 address. This feature enables the network to give a dynamic IPv6 address to a mobile node and authenticate the mobile node using authentication, authorization, and accounting (AAA). This option should be used when either Internet Key Exchange (IKE) or IPsec is not used for protecting BUs or binding acknowledgments (BAs).

In order to provide roaming services, a standardized method, such as NAI or a mobile node home address, is needed for identifying users. Roaming may be loosely defined as the ability to use any one of multiple Internet service providers (ISPs) while maintaining a formal, customer-vendor relationship with only one. Examples of where roaming capabilities might be required include ISP confederations and ISP-provided corporate network access support. Other entities interested in roaming capability may include the following:

- Regional ISPs, operating within a particular state or province, that want to combine efforts with those of other regional providers to offer dialup service over a wider area.
- National ISPs that want to combine their operations with those of one or more ISPs in another country to offer more comprehensive dialup service in a group of countries or on a continent.
- Wireless LAN hot spots that provide service to one or more ISPs.
- Businesses that want to offer their employees a comprehensive package of dialup services on a global basis. Those services may include Internet access and secure access to corporate intranets using a VPN.

Authentication Protocol for Mobile IPv6

The authentication protocol for Mobile IPv6 support secures mobile node and home agent signaling using the MN-HA mobility message authentication option, which authenticates the BU and BA messages based on the shared-key-based security association between the mobile node (MN) and the HA. This feature allows Mobile IPv6 to be deployed in a production environment where a non-IPsec authentication method is required. MN-HA consists of a mobility SPI, a shared key, an authentication algorithm, and the mobility message replay protection option.

The mobility SPI is a number from 256 through 4,294,967,296. The key consists of an arbitrary value and is 16 octets in length. The authentication algorithm used is HMAC_SHA1. The replay protection mechanism may use either the sequence number option or the time-stamp option. The MN-HA mobility message authentication option must be the last option in a message with a mobility header if it is the only mobility message authentication option in the message.

When a BU or BA message is received without the MN-HA option and the entity receiving it is configured to use the MN-HA option or has the shared-key-based mobility security association for the mobility message authentication option, the entity discards the received message.

The mobility message replay protection option allows the home agent to verify that a BU has been freshly generated by the mobile node and not replayed by an attacker from some previous BU. This functionality is

especially useful for cases where the home agent does not maintain stateful information about the mobile node after the binding entry has been removed. The home agent performs the replay protection check after the BU has been authenticated. The mobility message replay protection option is used by the mobile node for matching the BA with the BU. When the home agent receives the mobility message replay protection option in BU, it must include the mobility message replay protection option in the BA.

How to Implement Mobile IPv6

Enabling Mobile IPv6 on the Router

You can customize interface configuration parameters before you start Mobile IPv6 (see the Customizing Mobile IPv6 on the Interface, page 455) or while Mobile IPv6 is in operation.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 mobile home-agent** [**preference** *preference-value*
5. **exit**
6. **exit**
7. **show ipv6 mobile globals**
8. **show ipv6 mobile home-agent** *interface-type interface-number* [*prefix*]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface Ethernet 2`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**ipv6 mobile home-agent** [**preference** *preference-value* **Example:** `Router(config-if)# ipv6 mobile home-agent`	Initializes and starts the Mobile IPv6 home agent on a specific interface.
Step 5	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode, and returns the router to global configuration mode.
Step 6	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode, and returns the router to privileged EXEC mode.
Step 7	**show ipv6 mobile globals** **Example:** `Router# show ipv6 mobile globals`	Displays global Mobile IPv6 parameters.
Step 8	**show ipv6 mobile home-agent** *interface-type interface-number* [*prefix*]] **Example:** `Router# show ipv6 mobile home-agent`	Displays local and discovered neighboring home agents.

Configuring Binding Information for Mobile IPv6

Before you start Mobile IPv6 on a specified interface, you can configure binding information on the router.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 mobile home-agent**
4. **binding access** *access-list-name* | *auth-option* | *seconds* | *maximum* | *refresh*
5. **exit**
6. **exit**
7. **show ipv6 mobile binding** [**care-of-address** *address* | **home-address** *address* | *interface-type interface-number*
8. **show ipv6 mobile traffic**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 mobile home-agent** **Example:** `Router(config)# ipv6 mobile home-agent`	Places the router in home-agent configuration mode.
Step 4	**binding access** *access-list-name* | *auth-option* | *seconds* | *maximum* | *refresh* **Example:** `Router(config-ha)# binding`	Configures binding options for the Mobile IPv6 home agent feature.
Step 5	**exit** **Example:** `Router(config-ha)# exit`	Exits home-agent configuration mode, and returns the router to global configuration mode.

	Command or Action	Purpose
Step 6	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode, and returns the router to privileged EXEC mode.
Step 7	**show ipv6 mobile binding** [**care-of-address** *address* | **home-address** *address* | *interface-type interface-number* **Example:** `Router# show ipv6 mobile binding`	Displays information about the binding cache.
Step 8	**show ipv6 mobile traffic** **Example:** `Router# show ipv6 mobile traffic`	Displays information about BUs received and BAs sent.

Enabling and Configuring NEMO on the IPv6 Mobile Router

The NEMO basic support protocol enables mobile IPv6 networks to attach to different points in the Internet.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 mobile router**
4. **eui-interface** *interface-type interface-number*
5. **home-network** *ipv6-prefix*
6. **home-address** {**home-network** | *ipv6-address-identifier* | *interface*
7. **explicit-prefix**
8. **register** {**extend expire** *seconds* **retry** *number* **interval** *seconds* | **lifetime** *seconds* | **retransmit initial** *milliseconds* **maximum** *milliseconds* **retry** *number*}
9. **exit**
10. **exit**
11. **show ipv6 mobile router running-config** | **status**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 mobile router** **Example:** `Router(config)# ipv6 mobile router`	Enables IPv6 NEMO functionality on a router, and places the router in IPv6 mobile router configuration mode.
Step 4	**eui-interface** *interface-type interface-number* **Example:** `Router(IPv6-mobile-router)# eui-interface Ethernet0/0`	Uses the Media Access Control (MAC) address from a specified interface for deriving the IPv6 mobile home address.
Step 5	**home-network** *ipv6-prefix* **Example:** `Router(IPv6-mobile-router)# home-network 2001:0DB1:1/64`	Specifies the home network's IPv6 prefix on the mobile router. • Users can configure up to 10 home-network entries, and they are used in order of priority. The prefix identifies the home network of the mobile router and is used to discover when the mobile router is at home.
Step 6	**home-address** {**home-network** | *ipv6-address-identifier* | *interface* **Example:** `Router(IPv6-mobile-router)# home-address home-network eui-64`	Specifies the mobile router home address using an IPv6 address or interface identifier. • When multiple home networks have been configured, we recommend that you use the **home-address home-network** command syntax, so that the mobile router builds a home address that matches the home network to which it registers.
Step 7	**explicit-prefix** **Example:** `Router(IPv6-mobile-router)# explicit-prefix`	Registers IPv6 prefixes connected to the IPv6 mobile router.

	Command or Action	Purpose
Step 8	**register** {**extend expire** *seconds* **retry** *number* **interval** *seconds* | **lifetime** *seconds* | **retransmit initial** *milliseconds* **maximum** *milliseconds* **retry** *number*} **Example:** `Router(IPv6-mobile-router)# register lifetime 600`	Controls the registration parameters of the IPv6 mobile router.
Step 9	**exit** **Example:** `Router(IPv6-mobile-router)# exit`	Exits IPv6 mobile router configuration mode, and returns the router to global configuration mode.
Step 10	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode, and returns the router to privileged EXEC mode.
Step 11	**show ipv6 mobile router running-config** | **status**] **Example:** `Router# show ipv6 mobile router`	Displays configuration information and monitoring statistics about the IPv6 mobile router.

Enabling NEMO on the IPv6 Mobile Router Home Agent

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router nemo**
4. **distance** [*mobile-distance*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router nemo** **Example:** `Router(config)# ipv6 router nemo`	Enables the NEMO routing process on the home agent and place the router in router configuration mode.
Step 4	**distance** [*mobile-distance*] **Example:** `Router(config-rtr)# distance 10`	Defines an administrative distance for NEMO routes.

Enabling Roaming on the IPv6 Mobile Router Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 mobile router-service roam** [**bandwidth-efficient** | **cost-efficient** | **priority** *value*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface ethernet 0/0`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**ipv6 mobile router-service roam** [**bandwidth-efficient** | **cost-efficient** | **priority** *value*] **Example:** `Router(config-if)# ipv6 mobile router-service roam`	Enables the IPv6 mobile router interface to roam.

Filtering Mobile IPv6 Protocol Headers and Options

IPv6 extension headers have been developed to support the use of option headers specific to Mobile IPv6. The IPv6 mobility header, the type 2 routing header, and the destination option header allow the configuration of IPv6 access list entries that match Mobile-IPv6-specific ICMPv6 messages and allow the definition of entries to match packets that contain the new and modified IPv6 extension headers.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 access-list** *access-list-name*
4. **permit icmp** {*source-ipv6-prefix / prefix-length* | **any** | **host** *source-ipv6-address*| **auth**} [*operator port-number*]] {*destination-ipv6-prefix/prefix-length* | **any** | **host** *destination-ipv6-address* | **auth**} [*operator* [*port-number*]] [*icmp-type* [*icmp-code*] | *icmp-message*] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 access-list** *access-list-name* **Example:** `Router(config)# ipv6 access-list list1`	Defines an IPv6 access list and places the router in IPv6 access list configuration mode.

Command or Action	Purpose
Step 4 **permit icmp** {*source-ipv6-prefix* / *prefix-length* | **any** | **host** *source-ipv6-address*| **auth**} [*operator port-number*]] {*destination-ipv6-prefix/prefix-length* | **any** | **host** *destination-ipv6-address* | **auth**} [*operator* [*port-number*]] [*icmp-type* [*icmp-code*] | *icmp-message*] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name* **Example:** **Example:** or **Example:** **deny icmp** {*source-ipv6-prefix* / *prefix-length* | **any** | **host** *source-ipv6-address* | **auth**} [*operator port-number*]] {*destination-ipv6-prefix/prefix-length* | **any** | **host** *destination-ipv6-address* | **auth**} [*operator* [*port-number*]] [*icmp-type* [*icmp-code*] | *icmp-message*] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name* **Example:** `Router(config-ipv6-acl)# permit icmp host 2001:DB8:0:4::32 any routing-type 2` **Example:**	Specifies permit or deny conditions for Mobile-IPv6-specific option headers in an IPv6 access list. • The *icmp-type* argument can be (but is not limited to) one of the following Mobile-IPv6-specific options: ◦ dhaad-request--numeric value is 144 ◦ dhaad-reply--numeric value is 145 ◦ mpd-solicitation--numeric value is 146 ◦ mpd-advertisement--numeric value is 147 • When the **dest-option-type** keyword with the *doh-number* or *doh-type* argument is used, IPv6 packets are matched against the destination option extension header within each IPv6 packet header. • When the **mobility** keyword is used, IPv6 packets are matched against the mobility extension header within each IPv6 packet header. • When the **mobility-type** keyword with the *mh-number* or *mh-type* argument is used, IPv6 packets are matched against the mobility-type option extension header within each IPv6 packet header. • When the **routing-type** keyword and *routing-number* argument are used, IPv6 packets are matched against the routing-type option extension header within each IPv6 packet header.

Command or Action	Purpose
Example: or **Example:** `Router(config-ipv6-acl)# deny icmp host 2001:DB8:0:4::32 any routing-type 2`	

Controlling ICMP Unreachable Messages

When IPv6 is unable to route a packet, it generates an appropriate ICMP unreachable message directed toward the source of the packet. Perform this task to control ICMP unreachable messages for any packets arriving on a specified interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 unreachables**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface ethernet 0/0`	Specifies the interface type and number, and enters interface configuration mode.

	Command or Action	Purpose
Step 4	**ipv6 unreachables** **Example:** `Router(config-if)# ipv6 unreachables`	Enables the generation of ICMPv6 unreachable messages for any packets arriving on the specified interface.

Verifying Native IPv6 Tunneling for Mobile IPv6

Using the native IPv6 tunneling (or generic routing encapsulation [GRE]) infrastructure improves the scalability and switching performance of the home agent. After the home agent sends a BU from a mobile node, a tunnel interface is created with the encapsulation mode set to IPv6/IPv6, the source address set to that of the home agent address on the home interface of the mobile node, and the tunnel destination set to that of the CoA of the mobile node.

These features are transparent and need not be configured in order to work with Mobile IPv6. For further information on IPv6 tunneling and how to implement GRE tunneling in IPv6, see the *Implementing Tunneling for IPv6* module.

SUMMARY STEPS

1. **enable**
2. **show ipv6 mobile tunnels** [**summary** | **tunnel** *if-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ipv6 mobile tunnels** [**summary**	**tunnel** *if-number* **Example:** `Router# show ipv6 mobile tunnels`	Lists the Mobile IPv6 tunnels on the home agent.

Configuring and Verifying Host Groups for Mobile IPv6

Users can create mobile user or group policies using the host group configuration. The host group profile lookup interface will allow the lookup of the profile associated with the sender of the BU using the sender's profile name, IPv6 address, or NAI. The host profile lookup interface also specifies the authentication properties for the IPv6 mobile node by creating either a unidirectional or bidirectional SPI.

A mobile node can identify itself using its profile name or home address as an identifier, which the Mobile IPv6 protocol messages use as an identifier in their registration messages. However, for certain

deployments it is essential that the mobile node has the capability to identify itself using a logical identifier such as NAI rather than a network address.

Note

- You cannot configure two host group profiles with the same IPv6 address when using the IPv6 address option.
- You cannot configure a profile with the NAI option set to a realm name and the address option set to a specific IPv6 address. You can either remove the NAI option or specify a fully qualified user name for the NAI option.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 mobile home-agent**
4. **binding** [**access** *access-list-name* | *auth-option* | *seconds*| *maximum* | *refresh*
5. **host group** *profile-name*
6. **address** {*ipv6-address* | **autoconfig**
7. **nai realm** | **user** | **macaddress**] {*user* **@** *realm*| **@** *realm*
8. **authentication inbound-spi** {*hex-in* | **decimal** *decimal-in*} **outbound-spi** {*hex-out* | **decimal** *decimal-out*} | **spi** {*hex-value* | **decimal** *decimal-value*}} **key** {**ascii** *string* | **hex** *string*}[**algorithm** *algorithm-type*] [**replay within** *seconds*
9. **exit**
10. **exit**
11. **show ipv6 mobile host groups** *profile-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ipv6 mobile home-agent** **Example:** `Router(config)# ipv6 mobile home-agent`	Places the router in home-agent configuration mode.
Step 4	**binding** [**access** *access-list-name* | *auth-option* | *seconds*| *maximum* | *refresh* **Example:** `Router(config-ha)# binding 15`	Configures binding options for the Mobile IPv6 home agent feature.
Step 5	**host group** *profile-name* **Example:** `Router(config-ha)# host group profile1`	Creates a host configuration in Mobile IPv6. • Multiple instances with different profile names can be created and used.
Step 6	**address** {*ipv6-address* | **autoconfig** **Example:** `Router(config-ha)# address baba 2001:DB8:1`	Specifies the home address of the IPv6 mobile node.
Step 7	**nai realm** | **user** | **macaddress**] {*user* **@** *realm*| **@** *realm* **Example:** `Router(config-ha)# nai @cisco.com`	Specifies the NAI for the IPv6 mobile node.
Step 8	**authentication inbound-spi** {*hex-in* | **decimal** *decimal-in*} **outbound-spi** {*hex-out* | **decimal** *decimal-out*} | **spi** {*hex-value* | **decimal** *decimal-value*}} **key** {**ascii** *string* | **hex** *string*}[**algorithm** *algorithm-type*] [**replay within** *seconds* **Example:** `Router(config-ha)# authentication spi 500 key ascii cisco`	Specifies the authentication properties for the IPv6 mobile node by creating either a unidirectional or bidirectional SPI.
Step 9	**exit** **Example:** `Router(config-ha)# exit`	Exits home-agent configuration mode, and returns the router to global configuration mode.

	Command or Action	Purpose
Step 10	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode, and returns the router to privileged EXEC mode.
Step 11	**show ipv6 mobile host groups** *profile-name*] **Example:** `Router# show ipv6 mobile host groups`	Displays information about Mobile IPv6 host groups.

Customizing Mobile IPv6 on the Interface

Perform this task to customize interface configuration parameters for your router configuration. You can set these interface configuration parameters before you start Mobile IPv6 or while Mobile IPv6 is in operation. You can customize any of these parameters, as desired.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 mobile home-agent** [**preference** *preference-value*
5. **ipv6 nd advertisement-interval**
6. **ipv6 nd prefix** {*ipv6-prefix / prefix-length* | **default**} [[*valid-lifetime preferred-lifetime* | **at** *valid-date preferred-date*] | **infinite** | **no-advertise** | **off-link** | **no-rtr-address** | **no-autoconfig**
7. **ipv6 nd ra interval** {*maximum-secs* [*minimum-secs*] | **msec** *maximum-msecs* [*minimum-msecs*]}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Router(config)# interface serial 3`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**ipv6 mobile home-agent** [**preference** *preference-value* **Example:** `Router(config-if)# ipv6 mobile home-agent preference 10`	Configures the Mobile IPv6 home agent preference value on the interface.
Step 5	**ipv6 nd advertisement-interval** **Example:** `Router(config-if)# ipv6 nd advertisement-interval`	Configures the advertisement interval option to be sent in RAs.
Step 6	**ipv6 nd prefix** {*ipv6-prefix / prefix-length* | **default**} [[*valid-lifetime preferred-lifetime* | **at** *valid-date preferred-date*] | **infinite** | **no-advertise** | **off-link** | **no-rtr-address** | **no-autoconfig** **Example:** `Router(config-if)# ipv6 nd prefix 2001:DB8::/35 1000 900`	Configures which IPv6 prefixes are included in IPv6 RAs.
Step 7	**ipv6 nd ra interval** {*maximum-secs* [*minimum-secs*] | **msec** *maximum-msecs* [*minimum-msecs*]} **Example:** `Router(config-if)# ipv6 nd ra interval 201`	Configures the interval between IPv6 RA transmissions on an interface.

Monitoring and Maintaining Mobile IPv6 on the Router

SUMMARY STEPS

1. **enable**
2. **clear ipv6 mobile binding** [**care-of-address** *prefix* | **home-address** *prefix* | *interface type interface-number*
3. **clear ipv6 mobile home-agents** [*interface-type interface-number*
4. **clear ipv6 mobile traffic**
5. **debug ipv6 mobile binding-cache** | **forwarding** | **home-agent** | **registration**
6. **debug ipv6 mobile networks**
7. **debug ipv6 mobile router** [**detail**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ipv6 mobile binding** [**care-of-address** *prefix* | **home-address** *prefix* | *interface type interface-number* **Example:** `Router# clear ipv6 mobile binding`	Clears the Mobile IPv6 binding cache on a router.
Step 3	**clear ipv6 mobile home-agents** [*interface-type interface-number* **Example:** `Router# clear ipv6 mobile home-agents`	Clears the neighboring home agents list.
Step 4	**clear ipv6 mobile traffic** **Example:** `Router# clear ipv6 mobile traffic`	Clears the counters associated with Mobile IPv6.

	Command or Action	Purpose
Step 5	**debug ipv6 mobile binding-cache** | **forwarding** | **home-agent** | **registration** **Example:** `Router# debug ipv6 mobile registration`	Enables the display of debugging information for Mobile IPv6.
Step 6	**debug ipv6 mobile networks** **Example:** `Router# debug ipv6 mobile networks`	Displays debugging messages for IPv6 mobile networks.
Step 7	**debug ipv6 mobile router** [**detail**] **Example:** `Router# debug ipv6 mobile router`	Displays debugging messages for the IPv6 mobile router.

- Examples, page 458

Examples

Sample Output from the show ipv6 mobile binding Command

```
Router # show ipv6 mobile binding


Mobile IPv6 Binding Cache Entries:
2001:DB8:2000::1111/64
via care-of address 2001:DB8::A8BB:CCFF:FE01:F611
home-agent 2001:DB8:2000::2001
Prefix 2001:DB8:8000::/64
Prefix 2001:DB8:2000::1111/128
Prefix 2001:DB8:1000::1111/128 installed
state ACTIVE, sequence 23, flags AHRlK
lifetime: remaining 44 (secs), granted 60 (secs), requested 60 (secs)
interface Ethernet0/2
tunnel interface Tunnel0
0 tunneled, 0 reversed tunneled
Selection matched 1 bindings
```

Sample Output from the show ipv6 mobile globals Command

In the following example, the **show ipv6 mobile globals** command displays the binding parameters:

```
Router# show ipv6 mobile globals
Mobile IPv6 Global Settings:
  1 Home Agent service on following interfaces:
    Ethernet1/2
  Bindings:
```

```
    Maximum number is unlimited.
    1 bindings are in use
    1 bindings peak
    Binding lifetime permitted is 262140 seconds
    Recommended refresh time is 300 seconds
```

Sample Output from the show ipv6 mobile home-agent Command

In the following example, the fact that no neighboring mobile home agents were found is displayed:

```
Router# show ipv6 mobile home-agent
Home Agent information for Ethernet1/3
  Configured:
    FE80::20B:BFFF:FE33:501F
    preference 0 lifetime 1800
      global address 2001:DB8:1::2/64
  Discovered Home Agents:
    FE80::4, last update 0 min
    preference 0 lifetime 1800
      global address 2001:DB8:1::4/64
```

Sample Output from the show ipv6 mobile host groups Command

In the following example, information about a host group named localhost is displayed:

```
Router# show ipv6 mobile host groups
Mobile IPv6 Host Configuration
Mobile Host List:
Host Group Name: localhost
    NAI: sai@cisco.com
    Address: CAB:C0:CA5A:CA5A::CA5A
    Security Association Entry:
      SPI: (Hex: 501) (Decimal Int: 1281)
      Key Format: Hex    Key: baba
      Algorithm: HMAC_SHA1
      Replay Protection: On    Replay Window: 6 secs
```

Sample Output from the show ipv6 mobile router Command

The following example provides information about the IPv6 mobile router status when the router configured with IPv6 NEMO:

```
Router# show ipv6 mobile router
Mobile Reverse Tunnel established
--------------------------------
using Nemo Basic mode
Home Agent: 2001:DB8:2000::2001
CareOf Address: 2001:DB8::A8BB:CCFF:FE01:F611
Attachment Router: FE80::A8BB:CCFF:FE01:F511
Attachment Interface: Ethernet1/1
Home Network: 2001:DB8:2000:0:FDFF:FFFF:FFFF:FFFE/64
Home Address: 2001:DB8:2000::1111/64
```

Sample Output from the show ipv6 mobile traffic Command

In the following example, information about Mobile IPv6 traffic is displayed:

```
Router# show ipv6 mobile traffic

MIPv6 statistics:
  Rcvd: 6477 total
      0 truncated, 0 format errors
      0 checksum errors
    Binding Updates received:6477
      0 no HA option, 0 BU's length
      0 options' length, 0 invalid CoA
```

```
  Sent: 6477 generated
    Binding Acknowledgements sent:6477
      6477 accepted (0 prefix discovery required)
      0 reason unspecified, 0 admin prohibited
      0 insufficient resources, 0 home reg not supported
      0 not home subnet, 0 not home agent for node
      0 DAD failed, 0 sequence number
    Binding Errors sent:0
      0 no binding, 0 unknown MH
  Home Agent Traffic:
    6477 registrations, 0 deregistrations
    00:00:23 since last accepted HA registration
    unknown time since last failed HA registration
    unknown last failed registration code
    Traffic forwarded:
      0 tunneled, 0 reversed tunneled
    Dynamic Home Agent Address Discovery:
      1 requests received, 1 replies sent
    Mobile Prefix Discovery:
      0 solicitations received, 0 advertisements sent
```

Sample Output from the show ipv6 mobile tunnels Command

The following example displays information about the Mobile IPv6 tunnels on the home agent:

```
Router# show ipv6 mobile tunnels


Tunnel1:

 Source: 2001:0DB1:1:1

 Destination: 2001:0DB1:2:1

 Encapsulation Mode: IPv6/IPv6

 Egress Interface: Ethernet 1/0

 Switching Mode: Process

 Keep-Alive: Not Supported

 Path MTU Discovery: Enabled

 Input: 20 packets, 1200 bytes, 0 drops

 Output: 20 packets, 1200 bytes, 0 drops

 NEMO Options: Not Supported
```

Configuration Examples for Implementing Mobile IPv6

Example: Enabling Mobile IPv6 on the Router

The following example shows how to configure and enable Mobile IPv6 on a specified interface:

```
Router> enable

Router# config terminal

Router(config)# interface Ethernet 1

Router(config-if)# ipv6 mobile home-agent
```

Example: Enabling and Configuring NEMO on the IPv6 Mobile Router

The following example shows how to enable and configure NEMO on the IPv6 mobile router. The /128 subnet must be used; otherwise, the IPv6 mobile router will fail to register because it will believe the home network is locally connected:

```
ipv6 unicast-routing
!
interface ethernet0/0
no ip address
ipv6 address 2001:DB8:2000::1111/128
ipv6 nd ra mtu suppress
!
interface ethernet0/1
no ip address
ipv6 address 2001:DB8:1000::1111/128
ipv6 nd ra mtu suppress
!
interface Ethernet0/0
description Roaming Interface to AR2
no ip address
ipv6 address autoconfig
ipv6 enable
ipv6 nd ns-interval 5000
ipv6 mobile router-service roam
ipv6 rip home enable
!
interface Ethernet0/1
description Mobile Network Interface
no ip address
ipv6 address 2001:DB8:8000::8001/64
ipv6 enable
ipv6 nd advertisement-interval
ipv6 nd ra interval msec 1000
ipv6 rip home enable
!
interface Ethernet1/1
description Roaming Interface to AR1
no ip address
ipv6 address autoconfig
ipv6 enable
ipv6 nd ns-interval 5000
ipv6 mobile router-service roam priority 99
ipv6 rip home enable
!
ipv6 router rip home
!
ipv6 mobile router
host group mr-host-group
nai mr1@cisco.com
address 2001:DB8:2000::1112/128
authentication spi hex 100 key ascii hi
```

```
exit
home-network 2001:DB8:2000::/64 discover priority 127
home-network 2001:DB8:1000::/64 discover
home-address home-network eui-64
explicit-prefix
register lifetime 60
register retransmit initial 1000 maximum 1000 retry 1
register extend expire 20 retry 1 interval 1
```

Example: Enabling NEMO on the IPv6 Mobile Router Home Agent

The following example shows how to enable and configure NEMO on the IPv6 mobile router home agent. The anycast address is needed for DHAAD to work. The **redistribute nemo** command redistributes NEMO routes into the routing protocol:

```
ipv6 unicast-routing
!
interface Ethernet0/2
description To Network
no ip address
no ipv6 address
ipv6 address 2001:DB8:2000::2001/64
ipv6 address 2001:DB8:2000::FDFF:FFFF:FFFF:FFFE/64 anycast
ipv6 enable
ipv6 nd advertisement-interval
ipv6 nd ra lifetime 2
ipv6 nd ra interval msec 1000
ipv6 mobile home-agent preference 100
ipv6 mobile home-agent
ipv6 rip home enable
!
interface Ethernet2/2
description To CN2
no ip address
no ipv6 address
ipv6 address 2001:DB8:3000::3001/64
ipv6 enable
ipv6 rip home enable
!
ipv6 router nemo
!
ipv6 router rip home
redistribute nemo
poison-reverse
!
ipv6 mobile home-agent
host group mr-host-group
nai mr1@cisco.com
address 2001:DB8:2000::1112/64
authentication spi hex 100 key ascii hi
exit
host group mr2-host-group
nai mr2@cisco.com
address 2001:DB8:2000::2222
authentication spi decimal 512 key hex 12345678123456781234567812345678
exit
```

Example: Enabling Roaming on the IPv6 Mobile Router Interface

The following example shows how to enable roaming on the IPv6 mobile router interface:

```
Device(config)# interface ethernet 0/0
Device(config-if)# ipv6 mobile router-service roam
```

Example: Configuring Host Groups for Mobile IPv6

The following example shows how to configure a Mobile IPv6 host group named group1:

```
ipv6 mobile host group group1

  nai sri@cisco.com

  address autoconfig

  authentication spi 500 key ascii cisco
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 supported feature list	Start Here: Cisco IOS Software Release Specifics for IPv6 Features
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*
IPv6 simplified packet headers, IPv6 neighbor discovery, IPv6 stateless autoconfiguration, IPv6 stateful autoconfiguration	" Implementing IPv6 Addressing and Basic Connectivity " module of the *Cisco IOS IPv6 Configuration Guide*
IPv6 access lists	" Implementing Traffic Filters and Firewalls for IPv6 Security " module of the *Cisco IOS IPv6 Configuration Guide*
IPv6 tunneling	" Implementing Tunneling for IPv6 " module of the *Cisco IOS IPv6 Configuration Guide*
IPv4 mobility configuration and commands	• *Cisco IOS IP Mobility Configuration Guide* • *Cisco IOS IP Mobility Command Reference*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIBs	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 3775	*Mobility Support in IPv6*
RFC 3846	*Mobile IPv4 Extension for Carrying Network Access Identifiers*
RFC 3963	*Network Mobility (NEMO) Basic Support Protocol*
RFC 4282	*The Network Access Identifier*
RFC 4283	*Mobile Node Identifier Option for Mobile IPv6 (MIPv6)*
RFC 4285	*Authentication Protocol for Mobile IPv6*
RFC 4885	*Network Mobility Support Terminology*
RFC 4887	*Network Mobility Home Network Models*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing Mobile IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 21 Feature Information for Implementing Mobile IPv6

Feature Name	Releases	Feature Information
Mobile IPv6 Home Agent	12.3(14)T 12.4	The Mobile IPv6 feature uses the IPv6 address space to enable Mobile IP deployment in any kind of large environment. No foreign agent is needed to use Mobile IPv6.
IPv6 ACL Extensions for Mobile IPv6	12.4(2)T 12.2(33)SRB 12.2(33)SXI 15.0(1)S	IPv6 access lists can be configured to allow IPv6 access list entries matching Mobile-IPv6-specific ICMP messages to be configured and to allow the definition of entries to match packets containing Mobile IPv6 extension headers.
Mobile IP--Mobile IPv6 HA phase 2	12.4(11)T	This phase of development for Mobile IPv6 includes support for NAI, alternate authentication, and native IPv6 tunnel infrastructure.
Mobile Networks v6--Basic NEMO	12.4(20)T	The network mobility (NEMO) basic support protocol enables mobile IPv6 networks to attach to different points in the Internet.

Implementing IPv6 Multicast

Traditional IP communication allows a host to send packets to a single host (unicast transmission) or to all hosts (broadcast transmission). IPv6 multicast provides a third scheme, allowing a host to send a single data stream to a subset of all hosts (group transmission) simultaneously.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Implementing IPv6 Multicast

- In order to enable IPv6 multicast routing on a router, you must first enable IPv6 unicast routing on the router. For information on how to enable IPv6 unicast routing on a router, refer to Implementing IPv6 Addressing and Basic Connectivity.
- You must enable IPv6 unicast routing on all interfaces.
- This module assumes that you are familiar with IPv6 addressing and basic configuration. Refer to the Implementing IPv6 Addressing and Basic Connectivity module for more information.

Restrictions for Implementing IPv6 Multicast

- IPv6 multicast for Cisco IOS software uses MLD version 2. This version of MLD is fully backward-compatible with MLD version 1 (described in RFC 2710). Hosts that support only MLD version 1 will

interoperate with a router running MLD version 2. Mixed LANs with both MLD version 1 and MLD version 2 hosts are likewise supported.

- IPv6 multicast is supported only over IPv4 tunnels in Cisco IOS Release 12.3(2)T, Cisco IOS Release 12.2(18)S, and Cisco IOS Release 12.0(26)S.
- When the bidirectional (bidir) range is used in a network, all routers in that network must be able to understand the bidirectional range in the bootstrap message (BSM).
- IPv6 multicast routing is disabled by default when the **ipv6 unicast-routing** command is configured. On Cisco Catalyst 6500 and Cisco 7600 series routers, the **ipv6 multicast-routing** also must be enabled in order to use IPv6 unicast routing.

Platform-Specific Information and Restrictions

In Cisco IOS Release 12.0(26)S, IPv6 multicast is supported on the Cisco 12000 series Internet router only on the following line cards:

- IP Service Engine (ISE):
  - 4-port Gigabit Ethernet ISE
  - 4-port OC-3c/STM-1c POS/SDH ISE
  - 8-port OC-3c/STM-1c POS/SDH ISE
  - 16-port OC-3c/STM-1c POS/SDH ISE
  - 4-port OC-12c/STM-4c POS/SDH ISE
  - 1-port OC-48c/STM-16c POS/SDH ISE
- Engine 4 Plus (E4+) Packet-over-SONET (POS):
  - 4-port OC-48c/STM-16c POS/SDH
  - 1-port OC-192c/STM-64c POS/SDH

On Cisco 12000 series line cards, the IPv6 multicast feature includes support for Protocol Independent Multicast sparse mode (PIM-SM), Multicast Listener Discovery (MLDv2), static mroutes, and the IPv6 distributed Multicast Forwarding Information Base (MFIB).

Forwarding of IPv6 multicast traffic is hardware-based on Cisco 12000 series IP Service Engine (ISE) line cards that support IPv6 multicast and software-based on all other supported Cisco 12000 series line cards.

On Cisco 12000 series ISE line cards, IPv6 multicast is implemented so that if the number of IPv6 multicast routes exceeds the hardware capacity of the ternary content addressable memory (TCAM), the following error message is displayed to describe how to increase the TCAM hardware capacity for IPv6 multicast routes:

```
EE48-3-IPV6_TCAM_CAPACITY_EXCEEDED: IPv6 multicast pkts will be software switched.
To support more IPv6 multicast routes in hardware:
 Get current TCAM usage with: show controllers ISE <slot> tcam
 In config mode, reallocate TCAM regions e.g. reallocate Netflow TCAM to IPv6 Mcast
  hw-module slot <num> tcam carve rx_ipv6_mcast <v6-mcast-percent>
  hw-module slot <num> tcam carve rx_top_nf <nf-percent>
 Verify with show command that sum of all TCAM regions = 100%
 Reload the linecard for the new TCAM carve config to take effect
WARNING: Recarve may affect other input features(ACL,CAR,MQC,Netflow)
```

TCAM is used for IPv6 multicast forwarding lookups. To increase TCAM capacity for handling IPv6 multicast routes, you must use the **hw-module slot** *number* **tcam carve rx_ipv6_mcast** *v6-mcast-percentage*command in privileged EXEC mode, where *v6-mcast-percentage* specifies the percentage of TCAM hardware used by IPv6 multicast prefix.

For example, you can change the IPv6 multicast region from 1 percent (default) to 16 percent of the TCAM hardware by reallocating the NetFlow region from 35 percent (default) to 20 percent as follows:

```
Router# hw-module slot 3 tcam carve rx_ipv6_mcast 16
Router# hw-module slot 3 tcam carve rx_nf  20
```

On Cisco 12000 series router with IPv6 multicast enabled, if you delete a subinterface with IPv6 configured or if IPv6 is disabled on a subinterface, the associated main interface gets reset.

Note From Cisco IOS Release 12.0(32)SY11 and 12.0(33)S7, deleting a subinterface or disabling IPv6 on a subinterface will reset the associated main interface only if that subinterface is the last subinterface with IPv6 configured under the main interface.

IPv6 multicast hardware forwarding is support on the Cisco Catalyst 6500 and 7600 series in Cisco IOS Release 12.2(18)SXE.

Information About Implementing IPv6 Multicast

IPv6 Multicast Overview

An IPv6 multicast group is an arbitrary group of receivers that want to receive a particular data stream. This group has no physical or geographical boundaries--receivers can be located anywhere on the Internet or in any private network. Receivers that are interested in receiving data flowing to a particular group must join the group by signaling their local router. This signaling is achieved with the MLD protocol.

Routers use the MLD protocol to learn whether members of a group are present on their directly attached subnets. Hosts join multicast groups by sending MLD report messages. The network then delivers data to a potentially unlimited number of receivers, using only one copy of the multicast data on each subnet. IPv6 hosts that wish to receive the traffic are known as group members.

Packets delivered to group members are identified by a single multicast group address. Multicast packets are delivered to a group using best-effort reliability, just like IPv6 unicast packets.

The multicast environment consists of senders and receivers. Any host, regardless of whether it is a member of a group, can send to a group. However, only the members of a group receive the message.

A multicast address is chosen for the receivers in a multicast group. Senders use that address as the destination address of a datagram to reach all members of the group.

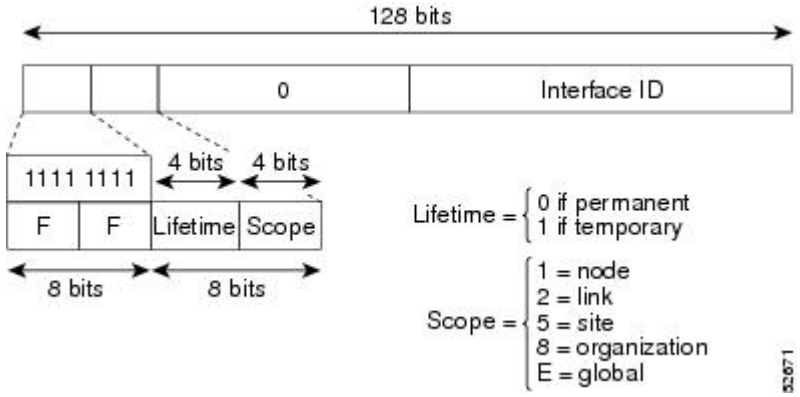
Membership in a multicast group is dynamic; hosts can join and leave at any time. There is no restriction on the location or number of members in a multicast group. A host can be a member of more than one multicast group at a time.

How active a multicast group is, its duration, and its membership can vary from group to group and from time to time. A group that has members may have no activity.

IPv6 Multicast Addressing

An IPv6 multicast address is an IPv6 address that has a prefix of FF00::/8 (1111 1111). An IPv6 multicast address is an identifier for a set of interfaces that typically belong to different nodes. A packet sent to a multicast address is delivered to all interfaces identified by the multicast address. The second octet following the prefix defines the lifetime and scope of the multicast address. A permanent multicast address has a lifetime parameter equal to 0; a temporary multicast address has a lifetime parameter equal to 1. A multicast address that has the scope of a node, link, site, or organization, or a global scope has a scope parameter of 1, 2, 5, 8, or E, respectively. For example, a multicast address with the prefix FF02::/16 is a permanent multicast address with a link scope. The figure below shows the format of the IPv6 multicast address.

Figure 32 IPv6 Multicast Address Format



IPv6 nodes (hosts and routers) are required to join (receive packets destined for) the following multicast groups:

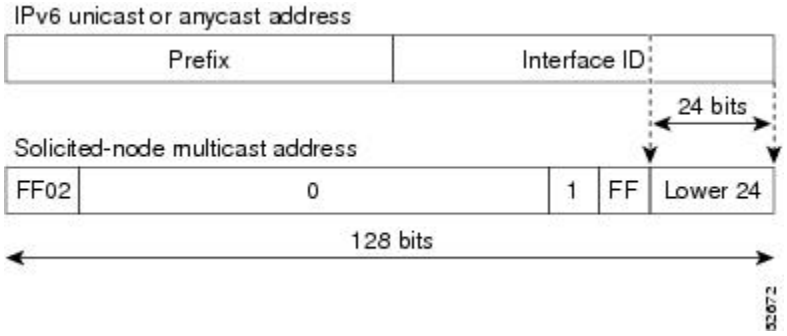
- All-nodes multicast group FF02:0:0:0:0:0:0:1 (scope is link-local)
- Solicited-node multicast group FF02:0:0:0:0:1:FF00:0000/104 for each of its assigned unicast and anycast addresses

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

The solicited-node multicast address is a multicast group that corresponds to an IPv6 unicast or anycast address. IPv6 nodes must join the associated solicited-node multicast group for every unicast and anycast address to which it is assigned. The IPv6 solicited-node multicast address has the prefix FF02:0:0:0:0:1:FF00:0000/104 concatenated with the 24 low-order bits of a corresponding IPv6 unicast or anycast address (see the figure below). For example, the solicited-node multicast address corresponding to

the IPv6 address 2037::01:800:200E:8C6C is FF02::1:FF0E:8C6C. Solicited-node addresses are used in neighbor solicitation messages.

Figure 33 IPv6 Solicited-Node Multicast Address Format



Note There are no broadcast addresses in IPv6. IPv6 multicast addresses are used instead of broadcast addresses.

- IPv6 Multicast Groups, page 471
- Scoped Address Architecture, page 471

IPv6 Multicast Groups

An IPv6 address must be configured on an interface before the interface can forward IPv6 traffic. Configuring a site-local or global IPv6 address on an interface automatically configures a link-local address and activates IPv6 for that interface. Additionally, the configured interface automatically joins the following required multicast groups for that link:

- Solicited-node multicast group FF02:0:0:0:0:1:FF00::/104 for each unicast and anycast address assigned to the interface

  Note The solicited-node multicast address is used in the neighbor discovery process.

- All-nodes link-local multicast group FF02::1
- All-routers link-local multicast group FF02::2

Scoped Address Architecture

IPv6 includes support for global and nonglobal addresses. This section describes the usage of IPv6 addresses of different scopes.

A scope zone, or a simply a zone, is a connected region of topology of a given scope. For example, the set of links connected by routers within a particular site, and the interfaces attached to those links, comprise a single zone of site-local scope.

A zone is a particular instance of a topological region (for example, Zone1's site or Zone2's site), whereas a scope is the size of a topological region (for example, a site or a link). The zone to which a particular nonglobal address pertains is not encoded in the address itself, but rather is determined by context, such as the interface from which it is sent or received. Therefore, addresses of a given nonglobal scope may be

reused in different zones of that scope. For example, Zone1’s site and Zone2’s site may each contain a node with site-local address FEC0::1.

Zones of the different scopes are instantiated as follows:

- Each link, and the interfaces attached to that link, comprises a single zone of link-local scope (for both unicast and multicast).
- There is a single zone of global scope (for both unicast and multicast), comprising all the links and interfaces in the Internet.
- The boundaries of zones of scope other than interface-local, link-local, and global must be defined and configured by network administrators. A site boundary serves as such for both unicast and multicast.

Zone boundaries are relatively static features and do not change in response to short-term changes in topology. Therefore, the requirement that the topology within a zone be "connected" is intended to include links and interfaces that may be connected only occasionally. For example, a residential node or network that obtains Internet access by dialup to an employer’s site may be treated as part of the employer’s site-local zone even when the dialup link is disconnected. Similarly, a failure of a router, interface, or link that causes a zone to become partitioned does not split that zone into multiple zones; rather, the different partitions are still considered to belong to the same zone.

Zones have the following additional properties:

- Zone boundaries cut through nodes, not links (the global zone has no boundary, and the boundary of an interface-local zone encloses just a single interface.)
- Zones of the same scope cannot overlap; that is, they can have no links or interfaces in common.
- A zone of a given scope (less than global) falls completely within zones of larger scope; that is, a smaller scope zone cannot include more topology than any larger scope zone with which it shares any links or interfaces.
- Each interface belongs to exactly one zone of each possible scope.

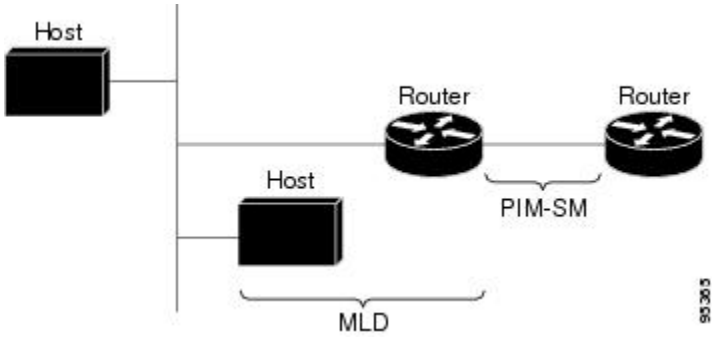
IPv6 Multicast Routing Implementation

Cisco software supports the following protocols to implement IPv6 multicast routing:

- MLD for IPv6. MLD is used by IPv6 routers to discover multicast listeners (nodes that want to receive multicast packets destined for specific multicast addresses) on directly attached links. There are two versions of MLD: MLD version 1 is based on version 2 of the Internet Group Management Protocol (IGMP) for IPv4, and MLD version 2 is based on version 3 of the IGMP for IPv4. IPv6 multicast for Cisco software uses both MLD version 2 and MLD version 1. MLD version 2 is fully backward-compatible with MLD version 1 (described in RFC 2710). Hosts that support only MLD version 1 will interoperate with a router running MLD version 2. Mixed LANs with both MLD version 1 and MLD version 2 hosts are likewise supported.
- PIM-SM is used between routers so that they can track which multicast packets to forward to each other and to their directly connected LANs.
- PIM in Source Specific Multicast (PIM-SSM) is similar to PIM-SM with the additional ability to report interest in receiving packets from specific source addresses (or from all but the specific source addresses) to an IP multicast address.

The figure below shows where MLD and PIM-SM operate within the IPv6 multicast environment.

Figure 34 IPv6 Multicast Routing Protocols Supported for IPv6



Multicast Listener Discovery Protocol for IPv6

To start implementing multicasting in the campus network, users must first define who receives the multicast. The MLD protocol is used by IPv6 routers to discover the presence of multicast listeners (for example, nodes that want to receive multicast packets) on their directly attached links, and to discover specifically which multicast addresses are of interest to those neighboring nodes. It is used for discovering local group and source-specific group membership. The MLD protocol provides a means to automatically control and limit the flow of multicast traffic throughout your network with the use of special multicast queriers and hosts.

The difference between multicast queriers and hosts is as follows:

- A querier is a network device, such as a router, that sends query messages to discover which network devices are members of a given multicast group.
- A host is a receiver, including routers, that send report messages to inform the querier of a host membership.

A set of queriers and hosts that receive multicast data streams from the same source is called a multicast group. Queriers and hosts use MLD reports to join and leave multicast groups and to begin receiving group traffic.

MLD uses the Internet Control Message Protocol (ICMP) to carry its messages. All MLD messages are link-local with a hop limit of 1, and they all have the router alert option set. The router alert option implies an implementation of the hop-by-hop option header.

MLD has three types of messages:

- Query--General, group-specific, and multicast-address-specific. In a query message, the multicast address field is set to 0 when MLD sends a general query. The general query learns which multicast addresses have listeners on an attached link.

Group-specific and multicast-address-specific queries are the same. A group address is a multicast address.

- Report--In a report message, the multicast address field is that of the specific IPv6 multicast address to which the sender is listening.
- Done--In a done message, the multicast address field is that of the specific IPv6 multicast address to which the source of the MLD message is no longer listening.

An MLD report must be sent with a valid IPv6 link-local source address, or the unspecified address (::), if the sending interface has not yet acquired a valid link-local address. Sending reports with the unspecified address is allowed to support the use of IPv6 multicast in the Neighbor Discovery Protocol.

For stateless autoconfiguration, a node is required to join several IPv6 multicast groups in order to perform duplicate address detection (DAD). Prior to DAD, the only address the reporting node has for the sending interface is a tentative one, which cannot be used for communication. Therefore, the unspecified address must be used.

MLD states that result from MLD version 2 or MLD version 1 membership reports can be limited globally or by interface. The MLD group limits feature provides protection against denial of service (DoS) attacks caused by MLD packets. Membership reports in excess of the configured limits will not be entered in the MLD cache, and traffic for those excess membership reports will not be forwarded.

MLD provides support for source filtering. Source filtering allows a node to report interest in listening to packets only from specific source addresses (as required to support SSM), or from all addresses except specific source addresses sent to a particular multicast address.

When a host using MLD version 1 sends a leave message, the router needs to send query messages to reconfirm that this host was the last MLD version 1 host joined to the group before it can stop forwarding traffic. This function takes about 2 seconds. This "leave latency" is also present in IGMP version 2 for IPv4 multicast.

MLD Access Group

The MLD access group provides receiver access control in Cisco IOS IPv6 multicast routers. This feature limits the list of groups a receiver can join, and it allows or denies sources used to join SSM channels.

Explicit Tracking of Receivers

The explicit tracking feature allows a router to track the behavior of the hosts within its IPv6 network. This feature also enables the fast leave mechanism to be used with MLD version 2 host reports.

IPv6 Multicast User Authentication and Profile Support

IPv6 multicast by design allows any host in the network to become a receiver or a source for a multicast group. Therefore, multicast access control is needed to control multicast traffic in the network. Access control functionality consists mainly of source access control and accounting, receiver access control and accounting, and provisioning of this access control mechanism.

Multicast access control provides an interface between multicast and authentication, authorization, and accounting (AAA) for provisioning, authorizing, and accounting at the last-hop router, receiver access control functions in multicast, and group or channel disabling capability in multicast.

When you deploy a new multicast service environment, it is necessary to add user authentication and provide a user profile download on a per-interface basis. The use of AAA and IPv6 multicast supports user authentication and downloading of the user profile in a multicast environment.

The event that triggers the download of a multicast access-control profile from the RADIUS server to the access router is arrival of an MLD join on the access router. When this event occurs, a user can cause the

authorization cache to time out and request download periodically or use an appropriate multicast clear command to trigger a new download in case of profile changes.

Accounting occurs via RADIUS accounting. Start and stop accounting records are sent to the RADIUS server from the access router. In order for you to track resource consumption on a per-stream basis, these accounting records provide information about the multicast source and group. The start record is sent when the last-hop router receives a new MLD report, and the stop record is sent upon MLD leave or if the group or channel is deleted for any reason.

IPv6 MLD Proxy

The MLD proxy feature provides a mechanism for a router to generate MLD membership reports for all (*, G)/(S, G) entries or a user-defined subset of these entries on the router's upstream interface. The MLD proxy feature enables a device to learn proxy group membership information, and forward multicast packets based upon that information.

If a router is acting as RP for mroute proxy entries, MLD membership reports for these entries can be generated on user specified proxy interface.

Protocol Independent Multicast

Protocol Independent Multicast (PIM) is used between routers so that they can track which multicast packets to forward to each other and to their directly connected LANs. PIM works independently of the unicast routing protocol to perform send or receive multicast route updates like other protocols. Regardless of which unicast routing protocols are being used in the LAN to populate the unicast routing table, Cisco IOS PIM uses the existing unicast table content to perform the Reverse Path Forwarding (RPF) check instead of building and maintaining its own separate routing table.

You can configure IPv6 multicast to use either PIM-SM or PIM-SSM operation, or you can use both PIM-SM and PIM-SSM together in your network.

- PIM-Sparse Mode, page 475
- IPv6 BSR: Configure RP Mapping, page 478
- PIM-Source Specific Multicast, page 478
- Routable Address Hello Option, page 481
- Bidirectional PIM, page 481

PIM-Sparse Mode

IPv6 multicast provides support for intradomain multicast routing using PIM-SM. PIM-SM uses unicast routing to provide reverse-path information for multicast tree building, but it is not dependent on any particular unicast routing protocol.

PIM-SM is used in a multicast network when relatively few routers are involved in each multicast and these routers do not forward multicast packets for a group, unless there is an explicit request for the traffic. PIM-SM distributes information about active sources by forwarding data packets on the shared tree. PIM-SM initially uses shared trees, which requires the use of an RP.

Requests are accomplished via PIM joins, which are sent hop by hop toward the root node of the tree. The root node of a tree in PIM-SM is the RP in the case of a shared tree or the first-hop router that is directly connected to the multicast source in the case of a shortest path tree (SPT). The RP keeps track of multicast groups and the hosts that send multicast packets are registered with the RP by that host's first-hop router.

As a PIM join travels up the tree, routers along the path set up multicast forwarding state so that the requested multicast traffic will be forwarded back down the tree. When multicast traffic is no longer

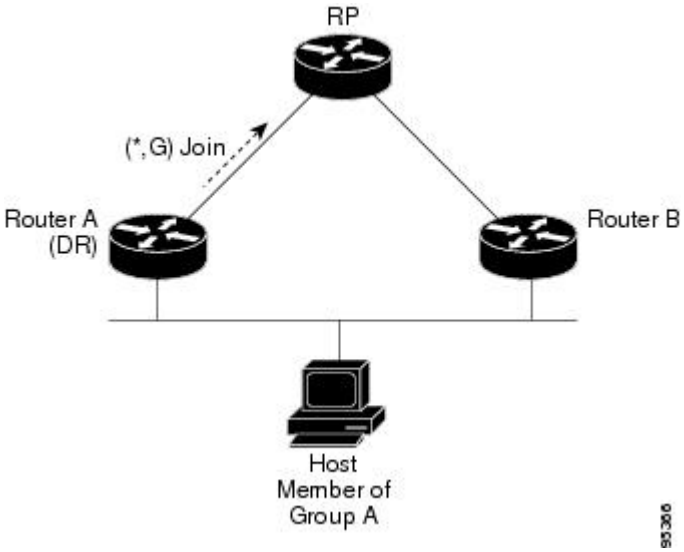
needed, a router sends a PIM prune up the tree toward the root node to prune (or remove) the unnecessary traffic. As this PIM prune travels hop by hop up the tree, each router updates its forwarding state appropriately. Ultimately, the forwarding state associated with a multicast group or source is removed.
A multicast data sender sends data destined for a multicast group. The designated router (DR) of the sender takes those data packets, unicast-encapsulates them, and sends them directly to the RP. The RP receives these encapsulated data packets, de-encapsulates them, and forwards them onto the shared tree. The packets then follow the (*, G) multicast tree state in the routers on the RP tree, being replicated wherever the RP tree branches, and eventually reaching all the receivers for that multicast group. The process of encapsulating data packets to the RP is called registering, and the encapsulation packets are called PIM register packets.

- Designated Router, page 476
- Rendezvous Point, page 477
- PIMv6 Anycast RP Solution Overview, page 478

Designated Router

Cisco routers use PIM-SM to forward multicast traffic and follow an election process to select a designated router when there is more than one router on a LAN segment.
The designated router is responsible for sending PIM register and PIM join and prune messages toward the RP to inform it about active sources and host group membership.
If there are multiple PIM-SM routers on a LAN, a designated router must be elected to avoid duplicating multicast traffic for connected hosts. The PIM router with the highest IPv6 address becomes the DR for the LAN unless you choose to force the DR election by use of the **ipv6 pim dr-priority** command. This command allows you to specify the DR priority of each router on the LAN segment (default priority = 1) so that the router with the highest priority will be elected as the DR. If all routers on the LAN segment have the same priority, then the highest IPv6 address is again used as the tiebreaker.
The figure below illustrates what happens on a multiaccess segment. Router A and Router B are connected to a common multiaccess Ethernet segment with Host A as an active receiver for Group A. Only Router A, operating as the DR, sends joins to the RP to construct the shared tree for Group A. If Router B was also permitted to send (*, G) joins to the RP, parallel paths would be created and Host A would receive duplicate multicast traffic. Once Host A begins to source multicast traffic to the group, the DR's responsibility is to send register messages to the RP. If both routers were assigned the responsibility, the RP would receive duplicate multicast packets.

Figure 35 Designated Router Election on a Multiaccess Segment



If the DR should fail, the PIM-SM provides a way to detect the failure of Router A and elect a failover DR. If the DR (Router A) became inoperable, Router B would detect this situation when its neighbor adjacency with Router A timed out. Because Router B has been hearing MLD membership reports from Host A, it already has MLD state for Group A on this interface and would immediately send a join to the RP when it became the new DR. This step reestablishes traffic flow down a new branch of the shared tree via Router B. Additionally, if Host A were sourcing traffic, Router B would initiate a new register process immediately after receiving the next multicast packet from Host A. This action would trigger the RP to join the SPT to Host A via a new branch through Router B.

Tip Two PIM routers are neighbors if there is a direct connection between them. To display your PIM neighbors, use the **show ipv6 pim neighbor** command in privileged EXEC mode.

Note The DR election process is required only on multiaccess LANs.

Rendezvous Point

IPv6 PIM provides embedded RP support. Embedded RP support allows the router to learn RP information using the multicast group destination address instead of the statically configured RP. For routers that are the RP, the router must be statically configured as the RP.

The router searches for embedded RP group addresses in MLD reports or PIM messages and data packets. On finding such an address, the router learns the RP for the group from the address itself. It then uses this learned RP for all protocol activity for the group. For routers that are the RP, the router is advertised as an embedded RP must be configured as the RP.

To select a static RP over an embedded RP, the specific embedded RP group range or mask must be configured in the access list of the static RP. When PIM is configured in sparse mode, you must also choose one or more routers to operate as an RP. An RP is a single common root placed at a chosen point of a shared distribution tree and is configured statically in each box.

PIM DRs forward data from directly connected multicast sources to the RP for distribution down the shared tree. Data is forwarded to the RP in one of two ways:

- Data is encapsulated in register packets and unicast directly to the RP by the first-hop router operating as the DR.
- If the RP has itself joined the source tree, it is multicast-forwarded per the RPF forwarding algorithm described in the PIM-Sparse Mode section.

The RP address is used by first-hop routers to send PIM register messages on behalf of a host sending a packet to the group. The RP address is also used by last-hop routers to send PIM join and prune messages to the RP to inform it about group membership. You must configure the RP address on all routers (including the RP router).

A PIM router can be an RP for more than one group. Only one RP address can be used at a time within a PIM domain for a certain group. The conditions specified by the access list determine for which groups the router is an RP.

IPv6 multicast supports the PIM accept register feature, which is the ability to perform PIM-SM register message filtering at the RP. The user can match an access list or compare the AS path for the registered source with the AS path specified in a route map.

PIMv6 Anycast RP Solution Overview

The anycast RP solution in IPv6 PIM allows an IPv6 network to support anycast services for the PIM-SM RP. It allows anycast RP to be used inside a domain that runs PIM only. Anycast RP can be used in IPv4 as well as IPv6, but it does not depend on the Multicast Source Discovery Protocol (MSDP), which runs only on IPv4. This feature is useful when interdomain connection is not required.

Anycast RP is a mechanism that ISP-based backbones use to get fast convergence when a PIM RP device fails. To allow receivers and sources to rendezvous to the closest RP, the packets from a source need to get to all RPs to find joined receivers.

A unicast IP address is chosen as the RP address. This address is either statically configured or distributed using a dynamic protocol to all PIM devices throughout the domain. A set of devices in the domain is chosen to act as RPs for this RP address; these devices are called the anycast RP set. Each device in the anycast RP set is configured with a loopback interface using the RP address. Each device in the anycast RP set also needs a separate physical IP address to be used for communication between the RPs.

The RP address, or a prefix that covers the RP address, is injected into the unicast routing system inside of the domain. Each device in the anycast RP set is configured with the addresses of all other devices in the anycast RP set, and this configuration must be consistent in all RPs in the set.

IPv6 BSR: Configure RP Mapping

PIM routers in a domain must be able to map each multicast group to the correct RP address. The BSR protocol for PIM-SM provides a dynamic, adaptive mechanism to distribute group-to-RP mapping information rapidly throughout a domain. With the IPv6 BSR feature, if an RP becomes unreachable, it will be detected and the mapping tables will be modified so that the unreachable RP is no longer used, and the new tables will be rapidly distributed throughout the domain.

Every PIM-SM multicast group needs to be associated with the IP or IPv6 address of an RP. When a new multicast sender starts sending, its local DR will encapsulate these data packets in a PIM register message and send them to the RP for that multicast group. When a new multicast receiver joins, its local DR will send a PIM join message to the RP for that multicast group. When any PIM router sends a (*, G) join message, the PIM router needs to know which is the next router toward the RP so that G (Group) can send a message to that router. Also, when a PIM router is forwarding data packets using (*, G) state, the PIM router needs to know which is the correct incoming interface for packets destined for G, because it needs to reject any packets that arrive on other interfaces.

A small set of routers from a domain are configured as candidate bootstrap routers (C-BSRs) and a single BSR is selected for that domain. A set of routers within a domain are also configured as candidate RPs (C-RPs); typically, these routers are the same routers that are configured as C-BSRs. Candidate RPs periodically unicast candidate-RP-advertisement (C-RP-Adv) messages to the BSR of that domain, advertising their willingness to be an RP. A C-RP-Adv message includes the address of the advertising C-RP, and an optional list of group addresses and mask length fields, indicating the group prefixes for which the candidacy is advertised. The BSR then includes a set of these C-RPs, along with their corresponding group prefixes, in bootstrap messages (BSMs) it periodically originates. BSMs are distributed hop-by-hop throughout the domain.

Bidirectional BSR support allows bidirectional RPs to be advertised in C-RP messages and bidirectional ranges in the BSM. All routers in a system must be able to use the bidirectional range in the BSM; otherwise, the bidirectional RP feature will not function.

PIM-Source Specific Multicast

PIM-SSM is the routing protocol that supports the implementation of SSM and is derived from PIM-SM. However, unlike PIM-SM where data from all multicast sources are sent when there is a PIM join, the SSM

feature forwards datagram traffic to receivers from only those multicast sources that the receivers have explicitly joined, thus optimizing bandwidth utilization and denying unwanted Internet broadcast traffic. Further, instead of the use of RP and shared trees, SSM uses information found on source addresses for a multicast group. This information is provided by receivers through the source addresses relayed to the last-hop routers by MLD membership reports, resulting in shortest-path trees directly to the sources.

In SSM, delivery of datagrams is based on (S, G) channels. Traffic for one (S, G) channel consists of datagrams with an IPv6 unicast source address S and the multicast group address G as the IPv6 destination address. Systems will receive this traffic by becoming members of the (S, G) channel. Signaling is not required, but receivers must subscribe or unsubscribe to (S, G) channels to receive or not receive traffic from specific sources.

MLD version 2 is required for SSM to operate. MLD allows the host to provide source information. Before SSM will run with MLD, SSM must be supported in the Cisco IPv6 router, the host where the application is running, and the application itself.

SSM Mapping for IPv6

SSM mapping for IPv6 supports both static and dynamic Domain Name System (DNS) mapping for MLD version 1 receivers. This feature allows deployment of IPv6 SSM with hosts that are incapable of providing MLD version 2 support in their TCP/IP host stack and their IP multicast receiving application.

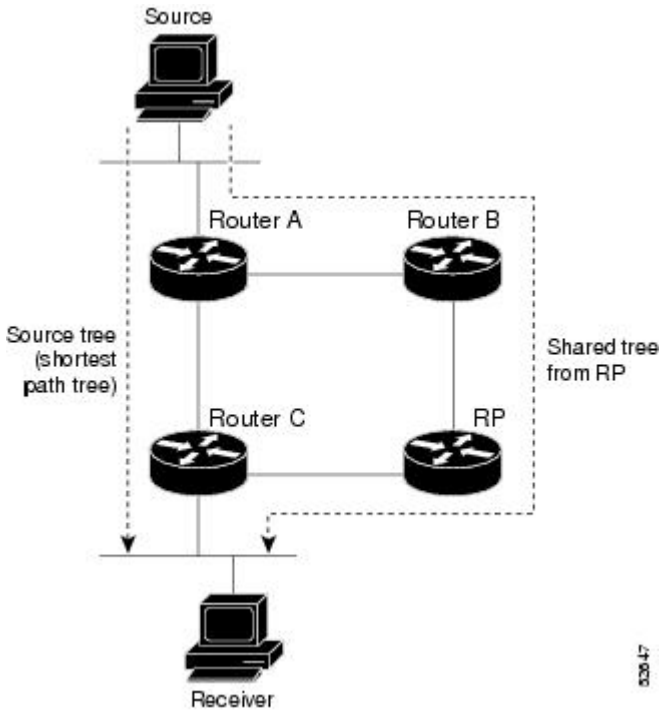
SSM mapping allows the router to look up the source of a multicast MLD version 1 report either in the running configuration of the router or from a DNS server. The router can then initiate an (S, G) join toward the source.

PIM Shared Tree and Source Tree (Shortest-Path Tree)

By default, members of a group receive data from senders to the group across a single data distribution tree rooted at the RP. This type of distribution tree is called shared tree or rendezvous point tree (RPT), as

illustrated in the figure below. Data from senders is delivered to the RP for distribution to group members joined to the shared tree.

Figure 36 Shared Tree and Source Tree (Shortest Path Tree)



If the data threshold warrants, leaf routers on the shared tree may initiate a switch to the data distribution tree rooted at the source. This type of distribution tree is called a shortest path tree or source tree. By default, the Cisco IOS software switches to a source tree upon receiving the first data packet from a source.

The following process details the move from shared tree to source tree:

1. Receiver joins a group; leaf Router C sends a join message toward the RP.
2. RP puts the link to Router C in its outgoing interface list.
3. Source sends the data; Router A encapsulates the data in the register and sends it to the RP.
4. RP forwards the data down the shared tree to Router C and sends a join message toward the source. At this point, data may arrive twice at Router C, once encapsulated and once natively.
5. When data arrives natively (unencapsulated) at the RP, the RP sends a register-stop message to Router A.
6. By default, receipt of the first data packet prompts Router C to send a join message toward the source.
7. When Router C receives data on (S, G), it sends a prune message for the source up the shared tree.
8. RP deletes the link to Router C from the outgoing interface of (S, G).
9. RP triggers a prune message toward the source.

Join and prune messages are sent for sources and RPs. They are sent hop-by-hop and are processed by each PIM router along the path to the source or RP. Register and register-stop messages are not sent hop-by-hop. They are sent by the designated router that is directly connected to a source and are received by the RP for the group.

Reverse Path Forwarding

Reverse-path forwarding is used for forwarding multicast datagrams. It functions as follows:

- If a router receives a datagram on an interface it uses to send unicast packets to the source, the packet has arrived on the RPF interface.
- If the packet arrives on the RPF interface, a router forwards the packet out the interfaces present in the outgoing interface list of a multicast routing table entry.
- If the packet does not arrive on the RPF interface, the packet is silently discarded to prevent loops.

PIM uses both source trees and RP-rooted shared trees to forward datagrams; the RPF check is performed differently for each, as follows:

- If a PIM router has source-tree state (that is, an (S, G) entry is present in the multicast routing table), the router performs the RPF check against the IPv6 address of the source of the multicast packet.
- If a PIM router has shared-tree state (and no explicit source-tree state), it performs the RPF check on the RP's address (which is known when members join the group).

Sparse-mode PIM uses the RPF lookup function to determine where it needs to send joins and prunes. (S, G) joins (which are source-tree states) are sent toward the source. (*, G) joins (which are shared-tree states) are sent toward the RP.

Routable Address Hello Option

When an IPv6 interior gateway protocol is used to build the unicast routing table, the procedure to detect the upstream router address assumes the address of a PIM neighbor is always same as the address of the next-hop router, as long as they refer to the same router. However, it may not be the case when a router has multiple addresses on a link.

Two typical situations can lead to this situation for IPv6. The first situation can occur when the unicast routing table is not built by an IPv6 interior gateway protocol such as multicast BGP. The second situation occurs when the address of an RP shares a subnet prefix with downstream routers (note that the RP router address has to be domain-wide and therefore cannot be a link-local address).

The routable address hello option allows the PIM protocol to avoid such situations by adding a PIM hello message option that includes all the addresses on the interface on which the PIM hello message is advertised. When a PIM router finds an upstream router for some address, the result of RPF calculation is compared with the addresses in this option, in addition to the PIM neighbor's address itself. Because this option includes all the possible addresses of a PIM router on that link, it always includes the RPF calculation result if it refers to the PIM router supporting this option.

Because of size restrictions on PIM messages and the requirement that a routable address hello option fits within a single PIM hello message, a limit of 16 addresses can be configured on the interface.

Bidirectional PIM

Bidirectional PIM allows multicast routers to keep reduced state information, as compared with unidirectional shared trees in PIM-SM. Bidirectional shared trees convey data from sources to the RPA and distribute them from the RPA to the receivers. Unlike PIM-SM, bidirectional PIM does not switch over to the source tree, and there is no register encapsulation of data from the source to the RP.

A single designated forwarder (DF) exists for each RPA on every link within a bidirectional PIM domain (including multiaccess and point-to-point links). The only exception is the RPL on which no DF exists. The DF is the router on the link with the best route to the RPA, which is determined by comparing MRIB-provided metrics. A DF for a given RPA forwards downstream traffic onto its link and forwards upstream

traffic from its link toward the rendezvous point link (RPL). The DF performs this function for all bidirectional groups that map to the RPA. The DF on a link is also responsible for processing Join messages from downstream routers on the link as well as ensuring that packets are forwarded to local receivers discovered through a local membership mechanism such as MLD.

Bidirectional PIM offers advantages when there are many moderate or low-rate sources. However, the bidirectional shared trees may have worse delay characteristics than do the source trees built in PIM-SM (depending on the topology).

Only static configuration of bidirectional RPs is supported in IPv6.

Static Mroutes

IPv6 static mroutes behave much in the same way as IPv4 static mroutes used to influence the RPF check. IPv6 static mroutes share the same database as IPv6 static routes and are implemented by extending static route support for RPF checks. Static mroutes support equal-cost multipath mroutes, and they also support unicast-only static routes.

MRIB

The Multicast Routing Information Base (MRIB) is a protocol-independent repository of multicast routing entries instantiated by multicast routing protocols (routing clients). Its main function is to provide independence between routing protocols and the Multicast Forwarding Information Base (MFIB). It also acts as a coordination and communication point among its clients.

Routing clients use the services provided by the MRIB to instantiate routing entries and retrieve changes made to routing entries by other clients. Besides routing clients, MRIB also has forwarding clients (MFIB instances) and special clients such as MLD. MFIB retrieves its forwarding entries from MRIB and notifies the MRIB of any events related to packet reception. These notifications can either be explicitly requested by routing clients or spontaneously generated by the MFIB.

Another important function of the MRIB is to allow for the coordination of multiple routing clients in establishing multicast connectivity within the same multicast session. MRIB also allows for the coordination between MLD and routing protocols.

MFIB

The MFIB is a platform-independent and routing-protocol-independent library for IPv6 software. Its main purpose is to provide a Cisco platform with an interface with which to read the IPv6 multicast forwarding table and notifications when the forwarding table changes. The information provided by the MFIB has clearly defined forwarding semantics and is designed to make it easy for the platform to translate to its specific hardware or software forwarding mechanisms.

When routing or topology changes occur in the network, the IPv6 routing table is updated, and those changes are reflected in the MFIB. The MFIB maintains next-hop address information based on the information in the IPv6 routing table. Because there is a one-to-one correlation between MFIB entries and routing table entries, the MFIB contains all known routes and eliminates the need for route cache maintenance that is associated with switching paths such as fast switching and optimum switching.

- Distributed MFIB, page 482

Distributed MFIB

Distributed Multicast Forwarding Information Base (MFIB) is used to switch multicast IPv6 packets on distributed platforms. Distributed MFIB may also contain platform-specific information on replication

across line cards. The basic MFIB routines that implement the core of the forwarding logic are common to all forwarding environments.

dMFIB implements the following functions:

- Distributes a copy of the MFIB to the line cards.
- Relays data-driven protocol events generated in the line cards to PIM.
- Provides an MFIB platform application program interface (API) to propagate MFIB changes to platform-specific code responsible for programming the hardware acceleration engine. This API also includes entry points to switch a packet in software (necessary if the packet is triggering a data-driven event) and to upload traffic statistics to the software.
- Provides hooks to allow clients residing on the RP to read traffic statistics on demand. Distributed MFIB does not periodically upload these statistics to the RP.

The combination of distributed MFIB and MRIB subsystems allows the device to have a "customized" copy of the MFIB database in each line card and to transport MFIB-related platform-specific information from the RP to the line cards.

IPv6 Multicast VRF Lite

The IPv6 Multicast VRF Lite feature provides IPv6 multicast support for multiple virtual routing/ forwarding contexts (VRFs). The scope of these VRFs is limited to the router in which the VRFs are defined.

This feature provides separation between routing and forwarding, providing an additional level of security because no communication between devices belonging to different VRFs is allowed unless it is explicitly configured. The IPv6 Multicast VRF Lite feature simplifies the management and troubleshooting of traffic belonging to a specific VRF.

IPv6 Multicast Process Switching and Fast Switching

A unified MFIB is used to provide both fast switching and process switching support for PIM-SM and PIM-SSM in IPv6 multicast. In process switching, the Route Processor must examine, rewrite, and forward each packet. The packet is first received and copied into the system memory. The router then looks up the Layer 3 network address in the routing table. The Layer 2 frame is then rewritten with the next-hop destination address and sent to the outgoing interface. The RP also computes the cyclic redundancy check (CRC). This switching method is the least scalable method for switching IPv6 packets.

IPv6 multicast fast switching allows routers to provide better packet forwarding performance than process switching. Information conventionally stored in a route cache is stored in several data structures for IPv6 multicast switching. The data structures provide optimized lookup for efficient packet forwarding.

In IPv6 multicast forwarding, the first packet is fast-switched if the PIM protocol logic allows it. In IPv6 multicast fast switching, the MAC encapsulation header is precomputed. IPv6 multicast fast switching uses the MFIB to make IPv6 destination prefix-based switching decisions. In addition to the MFIB, IPv6 multicast fast switching uses adjacency tables to prepend Layer 2 addressing information. The adjacency table maintains Layer 2 next-hop addresses for all MFIB entries.

The adjacency table is populated as adjacencies are discovered. Each time an adjacency entry is created (such as through ARP), a link-layer header for that adjacent node is precomputed and stored in the adjacency table. Once a route is determined, it points to a next hop and corresponding adjacency entry. It is subsequently used for encapsulation during switching of packets.

A route might have several paths to a destination prefix, such as when a router is configured for simultaneous load balancing and redundancy. For each resolved path, a pointer is added for the adjacency

corresponding to the next-hop interface for that path. This mechanism is used for load balancing across several paths.

Multiprotocol BGP for the IPv6 Multicast Address Family

The multiprotocol BGP for the IPv6 multicast address family feature provides multicast BGP extensions for IPv6 and supports the same features and functionality as IPv4 BGP. IPv6 enhancements to multicast BGP include support for an IPv6 multicast address family and network layer reachability information (NLRI) and next hop (the next router in the path to the destination) attributes that use IPv6 addresses.

Multicast BGP is an enhanced BGP that allows the deployment of interdomain IPv6 multicast. Multiprotocol BGP carries routing information for multiple network layer protocol address families; for example, IPv6 address family and for IPv6 multicast routes. The IPv6 multicast address family contains routes used for RPF lookup by the IPv6 PIM protocol, and multicast BGP IPv6 provides for interdomain transport of the same. Users must use multiprotocol BGP for IPv6 multicast when using IPv6 multicast with BGP because the unicast BGP learned routes will not be used for IPv6 multicast.

Multicast BGP functionality is provided through a separate address family context. A subsequent address family identifier (SAFI) provides information about the type of the network layer reachability information that is carried in the attribute. Multiprotocol BGP unicast uses SAFI 1 messages, and multiprotocol BGP multicast uses SAFI 2 messages. SAFI 1 messages indicate that the routes are usable only for IP unicast, not IP multicast. Because of this functionality, BGP routes in the IPv6 unicast RIB must be ignored in the IPv6 multicast RPF lookup.

A separate BGP routing table is maintained to configure incongruent policies and topologies (for example, IPv6 unicast and multicast) by using IPv6 multicast RPF lookup. Multicast RPF lookup is very similar to the IP unicast route lookup.

No MRIB is associated with the IPv6 multicast BGP table. However, IPv6 multicast BGP operates on the unicast IPv6 RIB when needed. Multicast BGP does not insert or update routes into the IPv6 unicast RIB.

NSF and SSO Support In IPv6 Multicast

Support for nonstop forwarding (NSF) and stateful switchover (SSO) is provided in IPv6 Multicast.

Bandwidth-Based CAC for IPv6 Multicast

The bandwidth-based call admission control (CAC) for IPv6 multicast feature implements a way to count per-interface mroute state limiters using cost multipliers. This feature can be used to provide bandwidth-based CAC on a per-interface basis in network environments where the multicast flows use different amounts of bandwidth.

This feature limits and accounts for IPv6 multicast state in detail. When this feature is configured, interfaces can be limited to the number of times they may be used as incoming or outgoing interfaces in the IPv6 multicast PIM topology.

With this feature, router administrators can configure global limit cost commands for state matching access lists and specify which cost multiplier to use when accounting such state against the interface limits. This feature provides the required flexibility to implement bandwidth-based local CAC policy by tuning appropriate cost multipliers for different bandwidth requirements.

How to Implement IPv6 Multicast

Enabling IPv6 Multicast Routing

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 multicast-routing** [**vrf** *vrf-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 multicast-routing** [**vrf** *vrf-name*] **Example:** `Router(config)# ipv6 multicast-routing`	Enables multicast routing on all IPv6-enabled interfaces and enables multicast forwarding for PIM and MLD on all enabled interfaces of the router.

Customizing and Verifying the MLD Protocol

Customizing and Verifying MLD on an Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 mld join-group** [*group-address*] [**include** | **exclude**] {*source-address* | **source-list** [*acl*]}
5. **ipv6 mld access-group** *access-list-name*
6. **ipv6 mld static-group** *group-address*] [**include**| **exclude**] {*source-address* | **source-list** [*acl*]}
7. **ipv6 mld query-max-response-time** *seconds*
8. **ipv6 mld query-timeout** *seconds*
9. **ipv6 mld query-interval** *seconds*
10. **exit**
11. **show ipv6 mld** [**vrf** *vrf-name*] **groups** [**link-local**] [*group-name* | *group-address*] [*interface-type interface-number*] [**detail** | **explicit**]
12. **show ipv6 mld groups summary**
13. **show ipv6 mld** [**vrf** *vrf-name*] **interface** [*type number*]
14. debug ipv6 mld [group-name | group-address | interface-type]
15. **debug ipv6 mld explicit** [*group-name* | *group-address*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Router(config)# interface FastEthernet 1/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**ipv6 mld join-group** [*group-address*] [**include** | **exclude**] {*source-address* | **source-list** [*acl*]} **Example:** `Router(config-if)# ipv6 mld join-group FF04::10`	Configures MLD reporting for a specified group and source.
Step 5	**ipv6 mld access-group** *access-list-name* **Example:** `Router(config-if)# ipv6 access-list acc-grp-1`	Allows the user to perform IPv6 multicast receiver access control.
Step 6	**ipv6 mld static-group** *group-address*] [**include**| **exclude**] {*source-address* | **source-list** [*acl*]} **Example:** `Router(config-if)# ipv6 mld static-group ff04::10 include 100::1`	Statically forwards traffic for the multicast group onto a specified interface and cause the interface to behave as if a MLD joiner were present on the interface.
Step 7	**ipv6 mld query-max-response-time** *seconds* **Example:** `Router(config-if)# ipv6 mld query-max-response-time 20`	Configures the maximum response time advertised in MLD queries.
Step 8	**ipv6 mld query-timeout** *seconds* **Example:** `Router(config-if)# ipv6 mld query-timeout 130`	Configures the timeout value before the router takes over as the querier for the interface.
Step 9	**ipv6 mld query-interval** *seconds* **Example:** `Router(config-if)# ipv6 mld query-interval 60`	Configures the frequency at which the Cisco IOS software sends MLD host-query messages. **Caution** Changing this value may severely impact multicast forwarding.

	Command or Action	Purpose
Step 10	**exit** **Example:** `Router(config-if)# exit`	Enter this command twice to exit interface configuration mode and enter privileged EXEC mode.
Step 11	**show ipv6 mld** [**vrf** *vrf-name*] **groups** [**link-local**] [*group-name* | *group-address*] [*interface-type interface-number*] [**detail** | **explicit**] **Example:** `Router# show ipv6 mld groups FastEthernet 2/1`	Displays the multicast groups that are directly connected to the router and that were learned through MLD.
Step 12	**show ipv6 mld groups summary** **Example:** `Router# show ipv6 mld groups summary`	Displays the number of (*, G) and (S, G) membership reports present in the MLD cache.
Step 13	**show ipv6 mld** [**vrf** *vrf-name*] **interface** [*type number*] **Example:** `Router# show ipv6 mld interface FastEthernet 2/1`	Displays multicast-related information about an interface.
Step 14	debug ipv6 mld [group-name | group-address | interface-type] **Example:** `Router# debug ipv6 mld`	Enables debugging on MLD protocol activity.
Step 15	**debug ipv6 mld explicit** [*group-name* | *group-address* **Example:** `Router# debug ipv6 mld explicit`	Displays information related to the explicit tracking of hosts.

Implementing MLD Group Limits

Per-interface and global MLD limits operate independently of each other. Both per-interface and global MLD limits can be configured on the same router. The number of MLD limits, globally or per interface, is not configured by default; the limits must be configured by the user. A membership report that exceeds either the per-interface or the global state limit is ignored.

- Implementing MLD Group Limits Globally, page 489

- Implementing MLD Group Limits per Interface, page 489

Implementing MLD Group Limits Globally

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 mld** [**vrf** *vrf-name*] **state-limit** *number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 mld** [**vrf** *vrf-name*] **state-limit** *number* **Example:** `Router(config)# ipv6 mld state-limit 300`	Limits the number of MLD states globally.

Implementing MLD Group Limits per Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 mld limit** *number* [**except** *access-list*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface FastEthernet 1/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**ipv6 mld limit** *number* [**except** *access-list* **Example:** `Router(config-if)# ipv6 mld limit 100`	Limits the number of MLD states on a per-interface basis.

Configuring Explicit Tracking of Receivers to Track Host Behavior

The explicit tracking feature allows a router to track the behavior of the hosts within its IPv6 network and enables the fast leave mechanism to be used with MLD version 2 host reports.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 mld explicit-tracking** *access-list-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface FastEthernet 1/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**ipv6 mld explicit-tracking** *access-list-name* **Example:** `Router(config-if)# ipv6 mld explicit-tracking list1`	Enables explicit tracking of hosts.

Configuring Multicast User Authentication and Profile Support

Prerequisites

Before you configure multicast user authentication and profile support, you may configure the following receiver access control functions in IPv6 multicast.

Restrictions

Before you configure multicast user authentication and profile support, you should be aware of the following restrictions:

- The port, interface, VC, or VLAN ID is the user or subscriber identity. User identity by hostname, user ID, or password is not supported.
- Enabling AAA Access Control for IPv6 Multicast, page 492

Enabling AAA Access Control for IPv6 Multicast

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **aaa new-model**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:** `Router(config)# aaa new-model`	Enables the AAA access control system.

Specifying Method Lists and Enabling Multicast Accounting

Perform this task to specify the method lists used for AAA authorization and accounting and how to enable multicast accounting on specified groups or channels on an interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **aaa authorization multicast default** [*method3* | *method4*
4. **aaa accounting multicast default** [**start-stop** | **stop-only**] [**broadcast**] [*method1*] [*method2*] [*method3*] [*method4*
5. **interface** *type number*
6. **ipv6 multicast aaa account receive** *access-list-name* [**throttle** *throttle-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**aaa authorization multicast default** [*method3* | *method4* **Example:** `Router(config)# aaa authorization multicast default`	Enables AAA authorization and sets parameters that restrict user access to an IPv6 multicast network.
Step 4	**aaa accounting multicast default** [**start-stop** | **stop-only**] [**broadcast**] [*method1*] [*method2*] [*method3*] [*method4* **Example:** `Router(config)# aaa accounting multicast default`	Enables AAA accounting of IPv6 multicast services for billing or security purposes when you use RADIUS.
Step 5	**interface** *type number* **Example:** `Router(config)# interface FastEthernet 1/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 6	**ipv6 multicast aaa account receive** *access-list-name* [**throttle** *throttle-number* **Example:** `Router(config-if)# ipv6 multicast aaa account receive list1`	Enables AAA accounting on specified groups or channels.

Disabling the Router from Receiving Unauthenticated Multicast Traffic

In some situations, access control may be needed to prevent multicast traffic from being received unless the subscriber is authenticated and the channels are authorized as per access control profiles. That is, there should be no traffic at all unless specified otherwise by access control profiles.

Perform this task to disable the router from receiving multicast traffic to be received from unauthenticated groups or unauthorized channels.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 multicast** [**vrf** *vrf-name*] **group-range**[*access-list-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 multicast** [**vrf** *vrf-name*] **group-range**[*access-list-name*] **Example:** `Router(config)# ipv6 multicast group-range`	Disables multicast protocol actions and traffic forwarding for unauthorized groups or channels on all the interfaces in a router.

Enabling MLD Proxy in IPv6

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 mld host-proxy** [*group-acl*
4. **ipv6 mld host-proxy interface** [*group-acl*]
5. **show ipv6 mld host-proxy** [*interface-type interface-number*] **group** [*group-address*]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 mld host-proxy** [*group-acl* **Example:** `Router(config)# ipv6 mld host-proxy proxy-group`	Enables the MLD proxy feature.
Step 4	**ipv6 mld host-proxy interface** [*group-acl*] **Example:** `Router(config)# ipv6 mld host-proxy interface Ethernet 0/0`	Enables the MLD proxy feature on a specified interface on an RP.
Step 5	**show ipv6 mld host-proxy** [*interface-type interface-number*] **group** [*group-address*]] **Example:** `Router# show ipv6 mld host-proxy Ethernet0/0`	Displays IPv6 MLD host proxy information.

- Resetting Authorization Status on an MLD Interface, page 495

Resetting Authorization Status on an MLD Interface

If no interface is specified, authorization is reset on all MLD interfaces.

SUMMARY STEPS

1. **enable**
2. **clear ipv6 multicast aaa authorization** [*interface-type interface-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ipv6 multicast aaa authorization** [*interface-type interface-number* **Example:** `Router# clear ipv6 multicast aaa authorization FastEthernet 1/0`	Clears parameters that restrict user access to an IPv6 multicast network.

Resetting the MLD Traffic Counters

SUMMARY STEPS

1. **enable**
2. **clear ipv6 mld** [**vrf** *vrf-name*] traffic
3. **show ipv6 mld** [**vrf** *vrf-name*] traffic

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ipv6 mld** [**vrf** *vrf-name*] traffic **Example:** `Router# clear ipv6 mld traffic`	Resets all MLD traffic counters.
Step 3	**show ipv6 mld** [**vrf** *vrf-name*] traffic **Example:** `Router# show ipv6 mld traffic`	Displays the MLD traffic counters.

Clearing the MLD Interface Counters

SUMMARY STEPS

1. **enable**
2. **clear ipv6 mld** [**vrf** *vrf-name*] **counters** *interface-type*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ipv6 mld** [**vrf** *vrf-name*] **counters** *interface-type* **Example:** `Router# clear ipv6 mld counters Ethernet1/0`	Clears the MLD interface counters.

Configuring PIM

Configuring PIM-SM and Displaying PIM-SM Information for a Group Range

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 pim** [**vrf** *vrf-name*] **rp-address** *ipv6-address* [*group-access-list*] [**bidir**]
4. **exit**
5. **show ipv6 pim** [**vrf** *vrf-name*] **interface** [**state-on**] [**state-off**] [*type number*]
6. **show ipv6 pim** [**vrf** *vrf-name*] **group-map** [*group-name* | *group-address*] | [*group-range* | *group-mask*] [**info-source** {**bsr** | **default** | **embedded-rp** | **static**}]
7. **show ipv6 pim** [**vrf** *vrf-name*] **neighbor** [**detail**] [*interface-type interface-number* | **count**]
8. **show ipv6 pim** [**vrf** *vrf-name*] **range-list**[**config**] [*rp-address* | *rp-name*]
9. **show ipv6 pim** [**vrf** *vrf-name*] **tunnel** [*interface-type interface-number*]
10. **debug ipv6 pim** [*group-name* | *group-address* | **interface** *interface-type* | **bsr** | **group** | **mvpn** | **neighbor**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 pim** [**vrf** *vrf-name*] **rp-address** *ipv6-address* [*group-access-list*] [**bidir**] **Example:** `Router(config)# ipv6 pim rp-address 2001:DB8::01:800:200E:8C6C acc-grp-1`	Configures the address of a PIM RP for a particular group range.
Step 4	**exit** **Example:** `Router(config-if)# exit`	Exits global configuration mode, and returns the router to privileged EXEC mode.

	Command or Action	Purpose
Step 5	**show ipv6 pim** [**vrf** *vrf-name*] **interface** [**state-on**] [**state-off**] [*type number*] **Example:** `Router# show ipv6 pim interface`	Displays information about interfaces configured for PIM.
Step 6	**show ipv6 pim** [**vrf** *vrf-name*] **group-map** [*group-name* | *group-address*] | [*group-range* | *group-mask*] [**info-source** {**bsr** | **default** | **embedded-rp** | **static**}] **Example:** `Router# show ipv6 pim group-map`	Displays an IPv6 multicast group mapping table.
Step 7	**show ipv6 pim** [**vrf** *vrf-name*] **neighbor** [**detail**] [*interface-type interface-number* | **count**] **Example:** `Router# show ipv6 pim neighbor`	Displays the PIM neighbors discovered by the Cisco IOS software.
Step 8	**show ipv6 pim** [**vrf** *vrf-name*] **range-list**[**config**] [*rp-address* | *rp-name*] **Example:** `Router# show ipv6 pim range-list`	Displays information about IPv6 multicast range lists.
Step 9	**show ipv6 pim** [**vrf** *vrf-name*] **tunnel** [*interface-type interface-number*] **Example:** `Router# show ipv6 pim tunnel`	Displays information about the PIM register encapsulation and de-encapsulation tunnels on an interface.
Step 10	**debug ipv6 pim** [*group-name* | *group-address* | **interface** *interface-type* | **bsr** | **group** | **mvpn** | **neighbor**] **Example:** `Router# debug ipv6 pim`	Enables debugging on PIM protocol activity.

Configuring PIM Options

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 pim** [**vrf** *vrf-name*] **spt-threshold infinity** [**group-list** *access-list-name*]
4. **ipv6 pim** [**vrf** *vrf-name*] **accept-register** {**list** *access-list* | **route-map** *map-name*}
5. **interface** *type number*
6. **ipv6 pim dr-priority** *value*
7. **ipv6 pim hello-interval** *seconds*
8. **ipv6 pim join-prune-interval** *seconds*
9. **exit**
10. **show ipv6 pim** [**vrf** *vrf-name*] **join-prune statistic** [*interface-type*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 pim** [**vrf** *vrf-name*] **spt-threshold infinity** [**group-list** *access-list-name*] **Example:** `Router(config)# ipv6 pim spt-threshold infinity group-list acc-grp-1`	Configures when a PIM leaf router joins the SPT for the specified groups.
Step 4	**ipv6 pim** [**vrf** *vrf-name*] **accept-register** {**list** *access-list* | **route-map** *map-name*} **Example:** `Router(config)# ipv6 pim accept-register route-map reg-filter`	Accepts or rejects registers at the RP.

	Command or Action	Purpose
Step 5	**interface** *type number* **Example:** `Router(config)# interface FastEthernet 1/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 6	**ipv6 pim dr-priority** *value* **Example:** `Router(config-if)# ipv6 pim dr-priority 3`	Configures the DR priority on a PIM router.
Step 7	**ipv6 pim hello-interval** *seconds* **Example:** `Router(config-if)# ipv6 pim hello-interval 45`	Configures the frequency of PIM hello messages on an interface.
Step 8	**ipv6 pim join-prune-interval** *seconds* **Example:** `Router(config-if)# ipv6 pim join-prune-interval 75`	Configures periodic join and prune announcement intervals for a specified interface.
Step 9	**exit** **Example:** `Router(config-if)# exit`	Enter this command twice to exit interface configuration mode and enter privileged EXEC mode.
Step 10	**show ipv6 pim** [**vrf** *vrf-name*] **join-prune statistic** [*interface-type*] **Example:** `Router# show ipv6 pim join-prune statistic`	Displays the average join-prune aggregation for the most recently aggregated packets for each interface.

Configuring Bidirectional PIM and Displaying Bidirectional PIM Information

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 pim** [**vrf** *vrf-name*] **rp-address** *ipv6-address* [*group-access-list*] [**bidir**]
4. **exit**
5. **show ipv6 pim** [**vrf** *vrf-name*] **df** [*interface-type interface-number*] [*rp-address*]
6. **show ipv6 pim** [**vrf** *vrf-name*] **df winner**[*interface-type interface-number*] [*rp-address*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 pim** [**vrf** *vrf-name*] **rp-address** *ipv6-address* [*group-access-list*] [**bidir**] **Example:** `Router(config)# ipv6 pim rp-address 2001:DB8::01:800:200E:8C6C bidir`	Configures the address of a PIM RP for a particular group range. Use of the **bidir** keyword means that the group range will be used for bidirectional shared-tree forwarding.
Step 4	**exit** **Example:** `Router(config-if)# exit`	Exits global configuration mode, and returns the router to privileged EXEC mode.
Step 5	**show ipv6 pim** [**vrf** *vrf-name*] **df** [*interface-type interface-number*] [*rp-address*] **Example:** `Router# show ipv6 pim df`	Displays the designated forwarder (DF)-election state of each interface for RP.

	Command or Action	Purpose
Step 6	**show ipv6 pim** [**vrf** *vrf-name*] **df winner**[*interface-type interface-number*] [*rp-address*] **Example:** `Router# show ipv6 pim df winner ethernet 1/0 200::1`	Displays the DF-election winner on each interface for each RP.

Resetting the PIM Traffic Counters

If PIM malfunctions or in order to verify that the expected number of PIM packets are received and sent, the user can clear PIM traffic counters. Once the traffic counters are cleared, the user can enter the **show ipv6 pim traffic** command to verify that PIM is functioning correctly and that PIM packets are being received and sent correctly.

SUMMARY STEPS

1. **enable**
2. **clear ipv6 pim** [**vrf** *vrf-name*] **traffic**
3. **show ipv6 pim** [**vrf** *vrf-name*] **traffic**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ipv6 pim** [**vrf** *vrf-name*] **traffic** **Example:** `Router# clear ipv6 pim traffic`	Resets the PIM traffic counters.
Step 3	**show ipv6 pim** [**vrf** *vrf-name*] **traffic** **Example:** `Router# show ipv6 pim traffic`	Displays the PIM traffic counters.

Clearing the PIM Topology Table to Reset the MRIB Connection

No configuration is necessary to use the MRIB. However, users may in certain situations want to clear the PIM topology table in order to reset the MRIB connection and verify MRIB information.

SUMMARY STEPS

1. **enable**
2. **clear ipv6 pim** [**vrf** *vrf-name*] **topology** [*group-name* | *group-address*]
3. **show ipv6 mrib** [**vrf** *vrf-name*] **client** [**filter**] [**name** {*client-name* | *client-name* **:** *client-id*}]
4. **show ipv6 mrib** [**vrf** *vrf-name*] **route** [**link-local**| **summary** | [*sourceaddress-or-name* | *****] [*groupname-or-address* [*prefix-length*]]]
5. **show ipv6 pim** [**vrf** *vrf-name*] **topology** [*groupname-or-address* [*sourcename-or-address*] | **link-local** | **route-count** [**detail**]]
6. **debug ipv6 mrib** [**vrf** *vrf-name*] **client**
7. **debug ipv6 mrib** [**vrf** *vrf-name*] **io**
8. **debug ipv6 mrib proxy**
9. **debug ipv6 mrib** [**vrf** *vrf-name*] **route** [*group-name* | *group-address*]
10. **debug ipv6 mrib** [**vrf** *vrf-name*] **table**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ipv6 pim** [**vrf** *vrf-name*] **topology** [*group-name*	*group-address*] **Example:** `Router# clear ipv6 pim topology FF04::10`	Clears the PIM topology table.
Step 3	**show ipv6 mrib** [**vrf** *vrf-name*] **client** [**filter**] [**name** {*client-name*	*client-name* **:** *client-id*}] **Example:** `Router# show ipv6 mrib client`	Displays multicast-related information about an interface.
Step 4	**show ipv6 mrib** [**vrf** *vrf-name*] **route** [**link-local**| **summary**	[*sourceaddress-or-name*	*****] [*groupname-or-address* [*prefix-length*]]] **Example:** `Router# show ipv6 mrib route`	Displays the MRIB route information.

	Command or Action	Purpose
Step 5	**show ipv6 pim** [**vrf** *vrf-name*] **topology** [*groupname-or-address* [*sourcename-or-address*] | **link-local** | **route-count** [**detail**]] **Example:** `Router# show ipv6 pim topology`	Displays PIM topology table information for a specific group or all groups.
Step 6	**debug ipv6 mrib** [**vrf** *vrf-name*] **client** **Example:** `Router# debug ipv6 mrib client`	Enables debugging on MRIB client management activity.
Step 7	**debug ipv6 mrib** [**vrf** *vrf-name*] **io** **Example:** `Router# debug ipv6 mrib io`	Enables debugging on MRIB I/O events.
Step 8	**debug ipv6 mrib proxy** **Example:** `Router# debug ipv6 mrib proxy`	Enables debugging on MRIB proxy activity between the route processor and line cards on distributed router platforms.
Step 9	**debug ipv6 mrib** [**vrf** *vrf-name*] **route** [*group-name* | *group-address*] **Example:** `Router# debug ipv6 mrib route`	Displays information about MRIB routing entry-related activity.
Step 10	**debug ipv6 mrib** [**vrf** *vrf-name*] **table** **Example:** `Router# debug ipv6 mrib table`	Enables debugging on MRIB table management activity.

Configuring a BSR

Configuring a BSR and Verifying BSR Information

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 pim** [**vrf** *vrf-name*] **bsr candidate bsr** *ipv6-address*[*hash-mask-length*] [**priority** *priority-value*]
4. **interface** *type number*
5. **ipv6 pim bsr border**
6. **exit**
7. **show ipv6 pim** [**vrf** *vrf-name*] **bsr** {**election** | **rp-cache** | **candidate-rp**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 pim** [**vrf** *vrf-name*] **bsr candidate bsr** *ipv6-address*[*hash-mask-length*] [**priority** *priority-value*] **Example:** `Router(config)# ipv6 pim bsr candidate bsr 2001:DB8:3000:3000::42 124 priority 10`	Configures a router to be a candidate BSR.
Step 4	**interface** *type number* **Example:** `Router(config)# interface FastEthernet 1/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 5	**ipv6 pim bsr border** **Example:** `Router(config-if)# ipv6 pim bsr border`	Configures a border for all BSMs of any scope on a specified interface.

	Command or Action	Purpose
Step 6	**exit** **Example:** `Router(config-if)# exit`	Enter this command twice to exit interface configuration mode and enter privileged EXEC mode.
Step 7	**show ipv6 pim** [**vrf** *vrf-name*] **bsr** {**election** | **rp-cache** | **candidate-rp**} **Example:** `Router# show ipv6 pim bsr election`	Displays information related to PIM BSR protocol processing.

Sending PIM RP Advertisements to the BSR

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 pim** [**vrf** *vrf-name*] **bsr candidate rp** *ipv6-address* [**group-list** *access-list-name*] [**priority** *priority-value*] [**interval** *seconds*] [**scope** *scope-value*] [**bidir**]
4. **interface** *type number*
5. **ipv6 pim bsr border**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ipv6 pim** [**vrf** *vrf-name*] **bsr candidate rp** *ipv6-address* [**group-list** *access-list-name*] [**priority** *priority-value*] [**interval** *seconds*] [**scope** *scope-value*] [**bidir**] **Example:** `Router(config)# ipv6 pim bsr candidate rp 2001:DB8:3000:3000::42 priority 0`	Sends PIM RP advertisements to the BSR.
Step 4	**interface** *type number* **Example:** `Router(config)# interface FastEthernet 1/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 5	**ipv6 pim bsr border** **Example:** `Router(config-if)# ipv6 pim bsr border`	Configures a border for all BSMs of any scope on a specified interface.

Configuring BSR for Use Within Scoped Zones

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 pim** [**vrf** *vrf-name*] **bsr candidate bsr** *ipv6-address* [*hash-mask-length*] [**priority** *priority-value*]
4. **ipv6 pim** [**vrf** *vrf-name*] **bsr candidate rp** *ipv6-address* [**group-list** *access-list-name*] [**priority** *priority-value*] [**interval** *seconds*] [**scope** *scope-value*] [**bidir**]
5. **interface** *type number*
6. **ipv6 multicast boundary scope** *scope-value*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 pim** [**vrf** *vrf-name*] **bsr candidate bsr** *ipv6-address* [*hash-mask-length*] [**priority** *priority-value*] **Example:** `Router(config)# ipv6 pim bsr candidate bsr 2001:DB8:1:1:4`	Configures a router to be a candidate BSR.
Step 4	**ipv6 pim** [**vrf** *vrf-name*] **bsr candidate rp** *ipv6-address* [**group-list** *access-list-name*] [**priority** *priority-value*] [**interval** *seconds*] [**scope** *scope-value*] [**bidir**] **Example:** `Router(config)# ipv6 pim bsr candidate rp 2001:DB8:1:1:1 group-list list scope 6`	Configures the candidate RP to send PIM RP advertisements to the BSR.
Step 5	**interface** *type number* **Example:** `Router(config)# interface FastEthernet 1/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 6	**ipv6 multicast boundary scope** *scope-value* **Example:** `Router(config-if)# ipv6 multicast boundary scope 6`	Configures a multicast boundary on the interface for a specified scope.

Configuring BSR Routers to Announce Scope-to-RP Mappings

IPv6 BSR routers can be statically configured to announce scope-to-RP mappings directly instead of learning them from candidate-RP messages. A user might want to configure a BSR router to announce scope-to-RP mappings so that an RP that does not support BSR is imported into the BSR. Enabling this feature also allows an RP positioned outside the enterprise's BSR domain to be learned by the known remote RP on the local candidate BSR routers.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 pim** [**vrf** *vrf-name*] **bsr announced rp** *ipv6-address* [**group-list** *access-list-name*] [**priority** *priority-value*] [**bidir**] [**scope** *scope-value*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 pim** [**vrf** *vrf-name*] **bsr announced rp** *ipv6-address* [**group-list** *access-list-name*] [**priority** *priority-value*] [**bidir**] [**scope** *scope-value*] **Example:** `Router(config)# ipv6 pim bsr announced rp 2001:DB8:3000:3000::42 priority 0`	Announces scope-to-RP mappings directly from the BSR for the specified candidate RP.

Configuring SSM Mapping

When the SSM mapping feature is enabled, DNS-based SSM mapping is automatically enabled, which means that the router will look up the source of a multicast MLD version 1 report from a DNS server.

You can use either DNS-based or static SSM mapping, depending on your router configuration. If you choose to use static SSM mapping, you can configure multiple static SSM mappings. If multiple static SSM mappings are configured, the source addresses of all matching access lists will be used.

Note To use DNS-based SSM mapping, the router needs to find at least one correctly configured DNS server, to which the router may be directly attached.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 mld** [**vrf** *vrf-name*] **ssm-map enable**
4. **no ipv6 mld** [**vrf** *vrf-name*] **ssm-map query dns**
5. **ipv6 mld** [**vrf** *vrf-name*] **ssm-map static** *access-list source-address*
6. **exit**
7. **show ipv6 mld** [**vrf** *vrf-name*] **ssm-map** [*source-address*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 mld** [**vrf** *vrf-name*] **ssm-map enable** **Example:** `Router(config)# ipv6 mld ssm-map enable`	Enables the SSM mapping feature for groups in the configured SSM range.
Step 4	**no ipv6 mld** [**vrf** *vrf-name*] **ssm-map query dns** **Example:** `Router(config)# no ipv6 mld ssm-map query dns`	Disables DNS-based SSM mapping.
Step 5	**ipv6 mld** [**vrf** *vrf-name*] **ssm-map static** *access-list source-address* **Example:** `Router(config)# ipv6 mld ssm-map static SSM_MAP_ACL_2 2001:DB8:1::1`	Configures static SSM mappings.

	Command or Action	Purpose
Step 6	**exit** **Example:** `Router(config-if)# exit`	Exits global configuration mode, and returns the router to privileged EXEC mode.
Step 7	**show ipv6 mld** [**vrf** *vrf-name*] **ssm-map** [*source-address*] **Example:** `Router# show ipv6 mld ssm-map`	Displays SSM mapping information.

Configuring Static Mroutes

Static multicast routes (mroutes) in IPv6 can be implemented as an extension of IPv6 static routes. You can configure your router to use a static route for unicast routing only, to use a static multicast route for multicast RPF selection only, or to use a static route for both unicast routing and multicast RPF selection.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 route** *ipv6-prefix / prefix-length ipv6-address* | *interface-type interface-number ipv6-address*]} [*administrative-distance*] [*administrative-multicast-distance* | **unicast**| **multicast**] [**tag** *tag*
4. **exit**
5. **show ipv6 mroute** [**vrf** *vrf-name*] [**link-local** | [*group-name* | *group-address* [*source-address* | *source-name*]] [**summary**] [**count**]
6. **show ipv6 mroute** [**vrf** *vrf-name*] [**link-local** | *group-name* | *group-address*] **active**[*kbps*]
7. **show ipv6 rpf** [**vrf** *vrf-name*] *ipv6-prefix*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 route** *ipv6-prefix / prefix-length ipv6-address | interface-type interface-number ipv6-address*]} [*administrative-distance*] [*administrative-multicast-distance* | **unicast**| **multicast**] [**tag** *tag* **Example:** `Router(config)# ipv6 route 2001:DB8::/64 6::6 100`	Establishes static IPv6 routes. The example shows a static route used for both unicast routing and multicast RPF selection.
Step 4	**exit** **Example:** `Router(config-if)# exit`	Exits global configuration mode, and returns the router to privileged EXEC mode.
Step 5	**show ipv6 mroute** [**vrf** *vrf-name*] [**link-local** | [*group-name* | *group-address* [*source-address* | *source-name*]] [**summary**] [**count**] **Example:** `Router# show ipv6 mroute ff07::1`	Displays the contents of the IPv6 multicast routing table.
Step 6	**show ipv6 mroute** [**vrf** *vrf-name*] [**link-local** | *group-name* | *group-address*] **active**[*kbps*] **Example:** `Router# show ipv6 mroute active`	Displays the active multicast streams on the router.
Step 7	**show ipv6 rpf** [**vrf** *vrf-name*] *ipv6-prefix* **Example:** `Router# show ipv6 rpf 2001:DB8::1:1:2`	Checks RPF information for a given unicast host address and prefix.

Configuring IPv6 Multiprotocol BGP

Configuring an IPv6 Peer Group to Perform Multicast BGP Routing

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **neighbor** *peer-group-name* **peer-group**
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
6. **address-family ipv6** [**unicast** | **multicast**]
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
8. **neighbor** {*ip-address* | *ipv6-address*} **peer-group** *peer-group-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified BGP routing process.

Command or Action	Purpose
Step 4 **neighbor** *peer-group-name* **peer-group** **Example:** `Device(config-router)# neighbor group1 peer-group`	Creates an multicast BGP peer group.
Step 5 **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 2001:DB8:0:CC00::1 remote-as 64600`	Adds the IPv6 address of the neighbor in the specified autonomous system to the IPv6 multicast BGP neighbor table of the local router. • The *ipv6-address*argument in the **neighbor remote-as**command must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons.
Step 6 **address-family ipv6** [**unicast** | **multicast**] **Example:** `Device(config-router)# address-family ipv6 multicast`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the router is placed in configuration mode for the IPv6 unicast address family if the **unicast** keyword is not specified with the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.
Step 7 **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 2001:DB8:0:CC00::1 activate`	Enables the neighbor to exchange prefixes for the specified family type with the neighbor and the local router. • To avoid extra configuration steps for each neighbor, use the **neighbor activate** command with the *peer-group-name* argument as an alternative in this step.
Step 8 **neighbor** {*ip-address* | *ipv6-address*} **peer-group** *peer-group-name* **Example:** `Device(config-router-af)# neighbor 2001:DB8:0:CC00::1 peer-group group1`	Assigns the IPv6 address of a BGP neighbor to a peer group.

- What to Do Next, page 515

What to Do Next

Refer to "Configuring an IPv6 Multiprotocol BGP Peer Group" in the Implementing Multiprotocol BGP for IPv6 document and the Cisco IOS IP Routing Configuration Guide for more information on assigning options to peer groups and making a BGP or multicast BGP neighbor a member of a peer group.

Advertising Routes into IPv6 Multiprotocol BGP

Perform this task to advertise (inject) a prefix into IPv6 multicast BGP.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv6** [**unicast** | **multicast**]
5. **network** *ipv6-address* / *prefix-length*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Router(config)# router bgp 65000`	Enters router configuration mode for the specified BGP routing process.
Step 4	**address-family ipv6** [**unicast** | **multicast**] **Example:** `Router(config-router)# address-family ipv6 multicast`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the router is placed in configuration mode for the IPv6 unicast address family if the **unicast** keyword is not specified with the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.

	Command or Action	Purpose
Step 5	**network** *ipv6-address* / *prefix-length* **Example:** `Router(config-router-af)# network 2001:DB8::/24`	Advertises (injects) the specified prefix into the IPv6 BGP database. (The routes must first be found in the IPv6 unicast routing table.) • Specifically, the prefix is injected into the database for the address family specified in the previous step. • Routes are tagged from the specified prefix as "local origin." • The *ipv6-prefix*argument in the **network**command must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons. • The *prefix-length* argument is a decimal value that indicates how many of the high-order contiguous bits of the address comprise the prefix (the network portion of the address). A slash mark must precede the decimal value.

Redistributing Prefixes into IPv6 Multiprotocol BGP

Redistribution is the process of redistributing, or injecting, prefixes from one routing protocol into another routing protocol. This task explains how to inject prefixes from a routing protocol into IPv6 multiprotocol BGP. Specifically, prefixes that are redistributed into IPv6 multiprotocol BGP using the **redistribute** router configuration command are injected into the IPv6 unicast database.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**]
5. **redistribute bgp** [*process-id*] [**metric** *metric-value*] [**route-map** *map-name*] [*source-protocol-options*]
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Command or Action	Purpose
Step 3 **router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified BGP routing process.
Step 4 **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**] **Example:** `Device(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the router is placed in configuration mode for the IPv6 unicast address family if a keyword is not specified with the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.
Step 5 **redistribute bgp** [*process-id*] [**metric** *metric-value*] [**route-map** *map-name*] [*source-protocol-options*] **Example:** `Device(config-router-af)# redistribute bgp 64500 metric 5 metric-type external`	Redistributes IPv6 routes from one routing domain into another routing domain.
Step 6 **exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode, and returns the router to router configuration mode. • Repeat this step to exit router configuration mode and return the router to global configuration mode.

Assigning a BGP Administrative Distance

Caution Changing the administrative distance of BGP internal routes is not recommended. One problem that can occur is the accumulation of routing table inconsistencies, which can break routing.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv6** [**unicast** | **multicast**}
5. **distance bgp** *external-distance internal-distance local-distance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 100`	Enters router configuration mode for the specified routing process.
Step 4	**address-family ipv6** [**unicast** | **multicast**} **Example:** `Device(config-router)# address-family ipv6 multicast`	Enters address family configuration mode for configuring routing sessions such as BGP that use standard IPv6 address prefixes.
Step 5	**distance bgp** *external-distance internal-distance local-distance* **Example:** `Device(config-router)# distance bgp 20 20 200`	Assigns a BGP administrative distance.

Generating Translate Updates for IPv6 Multicast BGP

The multicast BGP translate-update feature generally is used in a multicast BGP-capable router that peers with a customer site that has only a BGP-capable router; the customer site has not or cannot upgrade its router to a multicast BGP-capable image. Because the customer site cannot originate multicast BGP advertisements, the router with which it peers will translate the BGP prefixes into multicast BGP prefixes, which are used for multicast-source RPF lookup.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv6** [**unicast** | **multicast**}
5. **neighbor** *ipv6-address* **translate-update ipv6 multicast** [**unicast**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 100`	Enters router configuration mode for the specified routing process.
Step 4	**address-family ipv6** [**unicast**	**multicast**} **Example:** `Device(config-router)# address-family ipv6 multicast`	Enters address family configuration mode for configuring routing sessions such as BGP that use standard IPv6 address prefixes.
Step 5	**neighbor** *ipv6-address* **translate-update ipv6 multicast** [**unicast**] **Example:** `Device(config-router)# neighbor 2001:DB8:7000::2 translate-update ipv6 multicast`	Generates multiprotocol IPv6 BGP updates that correspond to unicast IPv6 updates received from a peer.

Resetting IPv6 BGP Sessions

SUMMARY STEPS

1. **enable**
2. **clear bgp ipv6** {**unicast** | **multicast**} {***** | *autonomous-system-number* | *ip-address* | *ipv6-address* | **peer-group** *peer-group-name*} [**soft**] [**in** | **out**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear bgp ipv6** {**unicast**	**multicast**} {*****	*autonomous-system-number*	*ip-address*	*ipv6-address*	**peer-group** *peer-group-name*} [**soft**] [**in**	**out**] **Example:** `Device# clear bgp ipv6 unicast peer-group marketing soft out`	Resets IPv6 BGP sessions.

Clearing External BGP Peers

SUMMARY STEPS

1. **enable**
2. **clear bgp ipv6** {**unicast** | **multicast**} **external** [**soft**] [**in** | **out**]
3. **clear bgp ipv6** {**unicast** | **multicast**} **peer-group** *name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear bgp ipv6** {**unicast**	**multicast**} **external** [**soft**] [**in**	**out**] **Example:** `Device# clear bgp ipv6 unicast external soft in`	Clears external IPv6 BGP peers.

	Command or Action	Purpose
Step 3	**clear bgp ipv6** {**unicast** | **multicast**} **peer-group** *name* **Example:** `Device# clear bgp ipv6 unicast peer-group marketing`	Clears all members of an IPv6 BGP peer group.

Clearing IPv6 BGP Route Dampening Information

SUMMARY STEPS

1. **enable**
2. **clear bgp ipv6** {**unicast** | **multicast**} **dampening** [*ipv6-prefix prefix-length*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear bgp ipv6** {**unicast** | **multicast**} **dampening** [*ipv6-prefix prefix-length*] **Example:** `Device# clear bgp ipv6 unicast dampening 2001:DB8::/64`	Clears IPv6 BGP route dampening information and unsuppresses the suppressed routes.

Clearing IPv6 BGP Flap Statistics

SUMMARY STEPS

1. **enable**
2. **clear bgp ipv6** {**unicast** | **multicast**} **flap-statistics** [*ipv6-prefix*/*prefix-length* | **regexp** *regexp* | **filter-list** *list*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear bgp ipv6** {**unicast** | **multicast**} **flap-statistics** [*ipv6-prefix*/*prefix-length* | **regexp** *regexp* | **filter-list** *list*] **Example:** `Device# clear bgp ipv6 unicast flap-statistics filter-list 3`	Clears IPv6 BGP flap statistics.

Configuring Bandwidth-Based CAC for IPv6

Configuring the Interface Limit for Bandwidth-Based CAC in IPv6

Bandwidth-based CAC for IPv6 counts per-interface IPv6 mroute states using cost multipliers. With this feature, router administrators can specify which cost multiplier to use when accounting such state against the interface limits.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits* / *prefix-length*}
5. **ipv6 multicast limit** [**connected** | **rpf** | **out**] *limit-acl max*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

Command or Action	Purpose
Step 2 **configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3 **interface** *type number* **Example:** `Router(config)# interface FastEthernet 1/3`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4 **ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits* / *prefix-length*} **Example:** `Router(config-if)# ipv6 address FE80::40:1:3 link-local`	Configures an IPv6 address based on an IPv6 general prefix.
Step 5 **ipv6 multicast limit** [**connected** | **rpf** | **out**] *limit-acl max* **Example:** `Router (config-if)# ipv6 multicast limit out acl1 10`	Configures per-interface mroute state limiters in IPv6.

Configuring an Access List for Bandwidth-Based CAC in IPv6

In bandwidth-based CAC for IPv6, router administrators can configure global limit cost commands for state matching access lists. Perform this task to configure an access list to configure a state matching access list.

or

deny

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 access-list** *access-list-name*
4. **permit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 access-list** *access-list-name* **Example:** `Router(config)# ipv6 access-list costlist1`	Defines an IPv6 access list and places the router in IPv6 access list configuration mode.
Step 4	**permit** **Example:** **Example:** or **Example:** **Example:** **deny** **Example:** `Router(config-ipv6-acl)# permit any ff03::1/64`	Use the **permit** or **deny** command to set conditions for an IPv6 access list.

Configuring the Global Limit for Bandwidth-Based CAC in IPv6

Router administrators can configure global limit cost commands for state matching access lists.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 multicast** [**vrf** *vrf-name*] **limit cost** *access-list cost-multiplier*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 multicast** [**vrf** *vrf-name*] **limit cost** *access-list cost-multiplier* **Example:** `Router (config)# ipv6 multicast limit cost costlist1 2`	Applies a cost to mroutes that match per-interface mroute state limiters in IPv6.

Using MFIB in IPv6 Multicast

Multicast forwarding is automatically enabled when IPv6 multicast routing is enabled.

- Verifying MFIB Operation in IPv6 Multicast, page 526
- Resetting MFIB Traffic Counters, page 528

Verifying MFIB Operation in IPv6 Multicast

Note Multicast forwarding is automatically enabled when IPv6 multicast routing is enabled.

SUMMARY STEPS

1. **enable**
2. **show ipv6 mfib** [**vrf** *vrf-name*] [**link-local** | **verbose** | *group-address-name* | *ipv6-prefix / prefix-length* | *source-address-name*| **active** | **count** | **interface** | **status** | **summary**]
3. **show ipv6 mfib** [**vrf** *vrf-name*] [**link-local**| *group-name* | *group-address*] **active** [*kbps*]
4. **show ipv6 mfib** [**vrf** *vrf-name*] [**all** | **linkscope**| *group-name* | *group-address* [*source-name* | *source-address*]] **count**
5. **show ipv6 mfib interface**
6. **show ipv6 mfib status**
7. **show ipv6 mfib** [**vrf** *vrf-name*] **summary**
8. **debug ipv6 mfib** [**vrf** *vrf-name*] [*group-name*| *group-address*] [**adjacency** | **db** | **fs** | **init** | **interface** | **mrib** [**detail**] | **nat** | **pak** | **platform** | **ppr** | **ps** | **signal** | **table**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ipv6 mfib** [**vrf** *vrf-name*] [**link-local**	**verbose**	*group-address-name*	*ipv6-prefix / prefix-length*	*source-address-name*| **active**	**count**	**interface**	**status**	**summary**] **Example:** `Device# show ipv6 mfib`	Displays the forwarding entries and interfaces in the IPv6 MFIB.
Step 3	**show ipv6 mfib** [**vrf** *vrf-name*] [**link-local**	*group-name*	*group-address*] **active** [*kbps*] **Example:** `Device# show ipv6 mfib active`	Displays the rate at which active sources are sending to multicast groups.
Step 4	**show ipv6 mfib** [**vrf** *vrf-name*] [**all**	**linkscope**	*group-name*	*group-address* [*source-name*	*source-address*]] **count** **Example:** `Device# show ipv6 mfib count`	Displays summary traffic statistics from the MFIB about the group and source.

	Command or Action	Purpose
Step 5	**show ipv6 mfib interface** **Example:** `Device# show ipv6 mfib interface`	Displays information about IPv6 multicast-enabled interfaces and their forwarding status.
Step 6	**show ipv6 mfib status** **Example:** `Device# show ipv6 mfib status`	Displays general MFIB configuration and operational status.
Step 7	**show ipv6 mfib** [**vrf** *vrf-name*] **summary** **Example:** `Device# show ipv6 mfib summary`	Displays summary information about the number of IPv6 MFIB entries and interfaces.
Step 8	**debug ipv6 mfib** [**vrf** *vrf-name*] [*group-name*| *group-address*] [**adjacency** | **db** | **fs** | **init** | **interface** | **mrib** [**detail**] | **nat** | **pak** | **platform** | **ppr** | **ps** | **signal** | **table**] **Example:** `Device# debug ipv6 mfib FF04::10 pak`	Enables debugging output on the IPv6 MFIB.

Resetting MFIB Traffic Counters

SUMMARY STEPS

1. **enable**
2. **clear ipv6 mfib** [**vrf** *vrf-name*] **counters** [*group-name* | *group-address* [*source-address* | *source-name*]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**clear ipv6 mfib** [**vrf** *vrf-name*] **counters** [*group-name* | *group-address* [*source-address* | *source-name*]] **Example:** `Device# clear ipv6 mfib counters FF04::10`	Resets all active MFIB traffic counters.

Disabling Default Features in IPv6 Multicast

Several features are automatically enabled when IPv6 multicast is used. However, a user may want to disable certain features in response to certain situations.

Disabling Embedded RP Support in IPv6 PIM

A user might want to disable embedded RP support on an interface if all of the routers in the domain do not support embedded RP.

Note This task disables PIM completely, not just embedded RP support in IPv6 PIM.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no ipv6 pim** [**vrf** *vrf-name*] **rp embedded**
4. **interface** *type number*
5. **no ipv6 pim**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**no ipv6 pim** [**vrf** *vrf-name*] **rp embedded** **Example:** `Router(config)# no ipv6 pim rp embedded`	Disables embedded RP support in IPv6 PIM.
Step 4	**interface** *type number* **Example:** `Router(config)# interface FastEthernet 1/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 5	**no ipv6 pim** **Example:** `Router(config-if)# no ipv6 pim`	Turns off IPv6 PIM on a specified interface.

Turning Off IPv6 PIM on a Specified Interface

A user might want only specified interfaces to perform IPv6 multicast and will therefore want to turn off PIM on a specified interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no ipv6 pim**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface FastEthernet 1/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**no ipv6 pim** **Example:** `Router(config-if)# no ipv6 pim`	Turns off IPv6 PIM on a specified interface.

Disabling MLD Router-Side Processing

A user might want only specified interfaces to perform IPv6 multicast and will therefore want to turn off MLD router-side processing on a specified interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no ipv6 mld router**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface FastEthernet 1/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**no ipv6 mld router** **Example:** `Router(config-if)# no ipv6 mld router`	Disables MLD router-side processing on a specified interface.

Disabling MFIB on the Router

Multicast forwarding is automatically enabled when IPv6 multicast routing is enabled. However, a user may want to disable multicast forwarding on the router.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no ipv6 mfib**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**no ipv6 mfib** **Example:** `Router(config)# no ipv6 mfib`	Disables IPv6 multicast forwarding on the router.

Disabling MFIB on a Distributed Platform

Multicast forwarding is automatically enabled when IPv6 multicast routing is enabled. However, you may want to disable multicast forwarding on a distributed platform.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 mfib-mode centralized-only**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 mfib-mode centralized-only** **Example:** `Device(config)# ipv6 mfib-mode centralized-only`	Disables distributed forwarding on a distributed platform.

Disabling MFIB Interrupt-Level IPv6 Multicast Forwarding

MFIB interrupt-level IPv6 multicast forwarding of outgoing packets on a specific interface is enabled on interfaces that support Cisco Express Forwarding. However, you may want to disable MFIB interrupt-level forwarding on a specified interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no ipv6 mfib cef output**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface FastEthernet 1/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**no ipv6 mfib cef output** **Example:** `Router(config-if)# no ipv6 mfib cef output`	Disables MFIB interrupt-level IPv6 multicast forwarding of outgoing packets on a specific interface.

Examples

This section provides the following command output examples:

Sample Output from the show ipv6 mfib Command

The following example displays the forwarding entries and interfaces in the MFIB. The router is configured for fast switching, and it has a receiver joined to FF05::1 on Ethernet1/1 and a source (2001:DB8:1:1:20) sending on Ethernet1/2:

```
Router# show ipv6 mfib
IP Multicast Forwarding Information Base
Entry Flags: C - Directly Connected, S - Signal, IA - Inherit A flag,
             AR - Activity Required, D - Drop
Forwarding Counts: Pkt Count/Pkts per second/Avg Pkt Size/Kbits per second
Other counts: Total/RPF failed/Other drops
Interface Flags: A - Accept, F - Forward, NS - Negate Signalling
             IC - Internal Copy, NP - Not platform switched
             SP - Signal Present
Interface Counts: FS Pkt Count/PS Pkt Count
(*,FF00::/8) Flags: C
   Forwarding: 0/0/0/0, Other: 0/0/0
   Tunnel0 Flags: NS
(*,FF00::/15) Flags: D
   Forwarding: 0/0/0/0, Other: 0/0/0
(*,FF05::1) Flags: C
   Forwarding: 2/0/100/0, Other: 0/0/0
   Tunnel0 Flags: A NS
   Ethernet1/1 Flags: F NS
     Pkts: 0/2
(2001:DB8:1:1:200,FF05::1) Flags:
   Forwarding: 5/0/100/0, Other: 0/0/0
   Ethernet1/2 Flags: A
   Ethernet1/1 Flags: F NS
     Pkts: 3/2
(*,FF10::/15) Flags: D
   Forwarding: 0/0/0/0, Other: 0/0/0
```

Sample Output from the show ipv6 mfib active Command

The following example displays statistics on the rate at which active IP multicast sources are sending information. The router is switching traffic from 2001:DB8:1:1:200 to FF05::1:

```
Router# show ipv6 mfib active
Active IPv6 Multicast Sources - sending >= 4 kbps
Group: FF05::1
  Source: 2001:DB8:1:1:200
    Rate: 20 pps/16 kbps(1sec), 0 kbps(last 128 sec)
```

Sample Output from the show ipv6 mfib count Command

The following example displays statistics from the MFIB about the group and source. The router is switching traffic from 2001:DB8:1:1:200 to FF05::1:

```
Router# show ipv6 mfib count

IP Multicast Statistics
54 routes, 7 groups, 0.14 average sources per group
Forwarding Counts: Pkt Count/Pkts per second/Avg Pkt Size/Kilobits per second
Other counts: Total/RPF failed/Other drops(OIF-null, rate-limit etc)
Group: FF00::/8
  RP-tree:    Forwarding: 0/0/0/0, Other: 0/0/0
Group: FF00::/15
  RP-tree:    Forwarding: 0/0/0/0, Other: 0/0/0
Group: FF05::1
  RP-tree:    Forwarding: 2/0/100/0, Other: 0/0/0
  Source: 10::1:1:200,   Forwarding: 367/10/100/7, Other: 0/0/0
  Tot. shown: Source count: 1, pkt count: 369
Group: FF10::/15
  RP-tree:    Forwarding: 0/0/0/0, Other: 0/0/0
Group: FF20::/15
  RP-tree:    Forwarding: 0/0/0/0, Other: 0/0/0
```

Sample Output from the show ipv6 mfib interface Command

The following example displays information about IPv6 multicast-enabled interfaces and their forwarding status. The router is configured for fast switching:

```
Router# show ipv6 mfib interface
IPv6 Multicast Forwarding (MFIB) status:
    Configuration Status: enabled
    Operational Status: running
MFIB interface       status    CEF-based output
                             [configured,available]
Ethernet1/1          up     [yes       ,yes      ]
Ethernet1/2          up     [yes       ,?        ]
Tunnel0              up     [yes       ,?        ]
Tunnel1              up     [yes       ,?        ]
```

Sample Output from the show ipv6 mfib summary Command

The following example displays summary information about the number of IPv6 MFIB entries and interfaces:

```
Router# show ipv6 mfib summary

IPv6 MFIB summary:
  54     total entries [1 (S,G), 7 (*,G), 46 (*,G/m)]
  17     total MFIB interfaces
```

Sample Output from the show ipv6 mld groups Command

The following is sample output from the **show ipv6 mld groups** command. It shows all of the groups joined by Fast Ethernet interface 2/1, including link-local groups used by network protocols.

```
Router# show ipv6 mld groups FastEthernet 2/1
MLD Connected Group Membership
Group Address          Interface           Uptime        Expires
FF02::2                FastEthernet2/1     3d18h         never
FF02::D                FastEthernet2/1     3d18h         never
FF02::16               FastEthernet2/1     3d18h         never
FF02::1:FF00:1         FastEthernet2/1     3d18h         00:00:27
FF02::1:FF00:79        FastEthernet2/1     3d18h         never
FF02::1:FF23:83C2      FastEthernet2/1     3d18h         00:00:22
FF02::1:FFAF:2C39      FastEthernet2/1     3d18h         never
FF06:7777::1           FastEthernet2/1     3d18h         00:00:26
```

Sample Output from the show ipv6 mld groups summary Command

The following is sample output from the **show ipv6 mld groups summary**command:

```
Router# show ipv6 mld groups summary
MLD Route Summary
  No. of (*,G) routes = 5
  No. of (S,G) routes = 0
```

Sample Output from the show ipv6 mld interface Command

The following is sample output from the **show ipv6 mld interface** command for Fast Ethernet interface 2/1:

```
Router# show ipv6 mld interface FastEthernet 2/1
FastEthernet2/1 is up, line protocol is up
Internet address is FE80::205:5FFF:FEAF:2C39/10
MLD is enabled in interface
Current MLD version is 2
MLD query interval is 125 seconds
```

```
MLD querier timeout is 255 seconds
MLD max query response time is 10 seconds
Last member query response interval is 1 seconds
MLD activity: 25 joins, 17 leaves
MLD querying router is FE80::205:5FFF:FEAF:2C39 (this system)
```

Sample Output from the show ipv6 mld ssm-map Command

The following examples show SSM mapping for the source address 2001:DB8::1:

```
Router# show ipv6 mld ssm-map 2001:DB8::1
 Group address  : 2001:DB8::1
 Group mode ssm : TRUE
 Database       : STATIC
 Source list    : 2001:DB8::2
                  2001:DB8::3
Router# show ipv6 mld ssm-map 2001:DB8::2
 Group address  : 2001:DB8::2
 Group mode ssm : TRUE
 Database       : DNS
 Source list    : 2001:DB8::3
                  2001:DB8::1
```

Sample Output from the show ipv6 mld traffic Command

The following example displays the MLD protocol messages received and sent:

```
Router# show ipv6 mld traffic

MLD Traffic Counters
Elapsed time since counters cleared:00:00:21
                              Received     Sent
Valid MLD Packets               3           1
Queries                         1           0
Reports                         2           1
Leaves                          0           0
Mtrace packets                  0           0
Errors:
Malformed Packets                           0
Bad Checksums                               0
Martian source                              0
Packets Received on MLD-disabled Interface  0
```

Sample Output from the show ipv6 mrib client Command

The following is sample output from the **show ipv6 mrib client**command:

```
Router# show ipv6 mrib client
IP MRIB client-connections
igmp:145        (connection id 0)
pim:146 (connection id 1)
mfib ipv6:3     (connection id 2)
slot 3  mfib ipv6 rp agent:16   (connection id 3)
slot 1  mfib ipv6 rp agent:16   (connection id 4)
slot 0  mfib ipv6 rp agent:16   (connection id 5)
slot 4  mfib ipv6 rp agent:16   (connection id 6)
slot 2  mfib ipv6 rp agent:16   (connection id 7)
```

Sample Output from the show ipv6 mrib route Command

The following is sample output from the **show ipv6 mrib route**command using the **summary** keyword:

```
Router# show ipv6 mrib route summary
MRIB Route-DB Summary
  No. of (*,G) routes = 52
  No. of (S,G) routes = 0
  No. of Route x Interfaces (RxI) = 10
```

Sample Output from the show ipv6 mroute Command

Using the **show ipv6 mroute** command is a good way to dynamically verify that multicast IPv6 data is flowing. The following is sample output from the **show ipv6 mroute**command:

```
Router# show ipv6 mroute ff07::1
Multicast Routing Table
Flags:D - Dense, S - Sparse, B - Bidir Group, s - SSM Group,
       C - Connected, L - Local, I - Received Source Specific Host Report,
       P - Pruned, R - RP-bit set, F - Register flag, T - SPT-bit set,
       J - Join SPT
Timers:Uptime/Expires
Interface state:Interface, State
(*, FF07::1), 00:04:45/00:02:47, RP 2001:DB8:6::6, flags:S
  Incoming interface:Tunnel5
  RPF nbr:6:6:6::6
  Outgoing interface list:
    POS4/0, Forward, 00:04:45/00:02:47
(2001:DB8:999::99, FF07::1), 00:02:06/00:01:23, flags:SFT
  Incoming interface:POS1/0
  RPF nbr:2001:DB8:999::99
  Outgoing interface list:
    POS4/0, Forward, 00:02:06/00:03:27
```

Sample Output from the show ipv6 mroute active Command

The following is sample output from the **show ipv6 mroute active**command:

```
Router# show ipv6 mroute active
Active IPv6 Multicast Sources - sending >= 4 kbps
Group:FF05::1
 Source:2001:DB8:1:1:1
   Rate:11 pps/8 kbps(1sec), 8 kbps(last 8 sec)
```

Sample Output from the show ipv6 pim bsr Command

The following example displays BSR election information:

```
Router# show ipv6 pim bsr election
PIMv2 BSR information
BSR Election Information
Scope Range List: ff00::/8
This system is the Bootstrap Router (BSR)
BSR Address: 2001:DB8:1:1:4
Uptime: 00:11:55, BSR Priority: 0, Hash mask length: 126
RPF: FE80::A8BB:CCFF:FE03:C400,Ethernet0/0
BS Timer: 00:00:07
This system is candidate BSR
Candidate BSR address: 2001:DB8:1:1:4, priority: 0, hash mask length: 126
```

Sample Output from the show ipv6 pim group-map Command

The following is sample output from the **show ipv6 pim group-map**command:

```
Router# show ipv6 pim group-map
FF33::/32*
      SSM
      Info source:Static
      Uptime:00:08:32, Groups:0
  FF34::/32*
      SSM
      Info source:Static
      Uptime:00:09:42, Groups:0
```

Sample Output from the show ipv6 pim interface Command

The following is sample output from the **show ipv6 pim interface** command using the **state-on** keyword:

```
Router# show ipv6 pim interface state-on
Interface          PIM  Nbr   Hello  DR
                        Count Intvl  Prior
Ethernet0          on   0     30     1
    Address:FE80::208:20FF:FE08:D7FF
    DR     :this system
POS1/0             on   0     30     1
    Address:FE80::208:20FF:FE08:D554
    DR     :this system
POS4/0             on   1     30   1
    Address:FE80::208:20FF:FE08:D554
    DR     :FE80::250:E2FF:FE8B:4C80
POS4/1             on   0     30   1
    Address:FE80::208:20FF:FE08:D554
    DR     :this system
Loopback0          on   0     30     1
    Address:FE80::208:20FF:FE08:D554
    DR     :this system
```

Sample Output from the show ipv6 pim join-prune statistic Command

The following example provides the join/prune aggregation on Ethernet interface 0/0/0:

```
Router# show ipv6 pim join-prune statistic Ethernet0/0/0
PIM Average Join/Prune Aggregation for last (1K/10K/50K) packets
Interface             Transmitted             Received
Ethernet0/0/0      0    / 0    / 0         1    / 0    / 0
```

Sample Output from the show ipv6 pim neighbor Command

The following is sample output from the **show ipv6 pim neighbor** command using the **detail** keyword to identify the additional addresses of the neighbors learned through the routable address hello option:

```
Router# show ipv6 pim neighbor detail
Neighbor Address(es)       Interface          Uptime    Expires DR pri Bidir
FE80::A8BB:CCFF:FE00:401   Ethernet0/0        01:34:16  00:01:16 1      B
60::1:1:3
FE80::A8BB:CCFF:FE00:501   Ethernet0/0        01:34:15  00:01:18 1      B
60::1:1:4
```

Sample Output from the show ipv6 pim range-list Command

The following is sample output from the **show ipv6 pim range-list**command:

```
Router# show ipv6 pim range-list
config SSM Exp:never Learnt from :::
 FF33::/32 Up:00:26:33
 FF34::/32 Up:00:26:33
 FF35::/32 Up:00:26:33
 FF36::/32 Up:00:26:33
 FF37::/32 Up:00:26:33
 FF38::/32 Up:00:26:33
 FF39::/32 Up:00:26:33
 FF3A::/32 Up:00:26:33
 FF3B::/32 Up:00:26:33
 FF3C::/32 Up:00:26:33
 FF3D::/32 Up:00:26:33
 FF3E::/32 Up:00:26:33
 FF3F::/32 Up:00:26:33
config SM RP:40::1:1:1 Exp:never Learnt from :::
 FF13::/64 Up:00:03:50
config SM RP:40::1:1:3 Exp:never Learnt from :::
 FF09::/64 Up:00:03:50
```

Sample Output from the show ipv6 pim topology Command

The following is sample output from the **show ipv6 pim topology** command:

```
Router# show ipv6 pim topology
IP PIM Multicast Topology Table
Entry state:(*/S,G)[RPT/SPT] Protocol Uptime Info
Entry flags:KAT - Keep Alive Timer, AA - Assume Alive, PA - Probe Alive,
    RA - Really Alive, LH - Last Hop, DSS - Don't Signal Sources,
    RR - Register Received, SR - Sending Registers, E - MSDP External,
    DCC - Don't Check Connected
Interface state:Name, Uptime, Fwd, Info
Interface flags:LI - Local Interest, LD - Local Dissinterest,
II - Internal Interest, ID - Internal Dissinterest,
LH - Last Hop, AS - Assert, AB - Admin Boundary
(*,FF05::1)
SM UP:02:26:56 JP:Join(now) Flags:LH
RP:2001:DB8:1:1:2
RPF:Ethernet1/1,FE81::1
  Ethernet0/1          02:26:56  fwd LI LH
(2001:DB8:1:1:200,FF05::1)
SM UP:00:00:07 JP:Null(never) Flags:
RPF:Ethernet1/1,FE80::30:1:4
  Ethernet1/1          00:00:07  off LI
```

Sample Output from the show ipv6 pim traffic Command

The following example shows the number of PIM protocol messages received and sent.

```
Router# show ipv6 pim traffic

PIM Traffic Counters
Elapsed time since counters cleared:00:05:29
                              Received     Sent
Valid PIM Packets               22          22
Hello                           22          22
Join-Prune                      0           0
Register                        0           0
Register Stop                   0           0
Assert                          0           0
Bidir DF Election               0           0
Errors:
Malformed Packets                           0
Bad Checksums                               0
Send Errors                                 0
Packet Sent on Loopback Errors              0
Packets Received on PIM-disabled Interface  0
Packets Received with Unknown PIM Version   0
```

Sample Output from the show ipv6 pim tunnel Command

The following is sample output from the **show ipv6 pim tunnel**command on the RP:

```
Router# show ipv6 pim tunnel
Tunnel0*
 Type  :PIM Encap
 RP    :100::1
 Source:100::1
Tunnel0*
 Type  :PIM Decap
 RP    :100::1
 Source: -
```

The following is sample output from the **show ipv6 pim tunnel**command on a non-RP:

```
Router# show ipv6 pim tunnel
Tunnel0*
 Type  :PIM Encap
```

```
 RP    :100::1
 Source:2001::1:1:1
```

Sample Output from the show ipv6 rpf Command

The following example displays RPF information for the unicast host with the IPv6 address of 2001:DB8:1:1:2:

```
Router# show ipv6 rpf 2001:DB8:1:1:2
RPF information for 2001:DB8:1:1:2
  RPF interface:Ethernet3/2
  RPF neighbor:FE80::40:1:3
  RPF route/mask:20::/64
  RPF type:Unicast
  RPF recursion count:0
  Metric preference:110
  Metric:30
```

Configuration Examples for Implementing IPv6 Multicast

Example: Enabling IPv6 Multicast Routing

The following example enables multicast routing on all interfaces. Entering this command also enables multicast forwarding for PIM and MLD on all enabled interfaces of the router.

```
Router> enable
Router# configure terminal

Router(config)# ipv6 multicast-routing
```

Example: Configuring the MLD Protocol

The following example shows how to configure the query maximum response time, the query timeout, and the query interval on FastEthernet interface 1/0:

```
Router> enable
```

```
Router# configure terminal
Router(config)# interface FastEthernet 1/0

Router(config-if)# ipv6 mld query-max-response-time  20

Router(config-if)# ipv6 mld query-timeout 130

Router(config-if)# ipv6 mld query-interval 60
```

The following example shows how to configure MLD reporting for a specified group and source, allows the user to perform IPv6 multicast receiver access control, and statically forwards traffic for the multicast group onto FastEthernet interface 1/0:

```
Router> enable
Router# configure terminal
Router(config)# interface FastEthernet 1/0
Router(config)# ipv6 mld join-group FF04::10
Router(config)# ipv6 mld static-group FF04::10 100::1
Router(config)# ipv6 mld access-group acc-grp-1
```

The following example shows information from the **show ipv6 mld interface** command for Fast Ethernet interface 2/1:

```
Router# show ipv6 mld interface FastEthernet 2/1

FastEthernet2/1 is up, line protocol is up
Internet address is FE80::205:5FFF:FEAF:2C39/10
MLD is enabled in interface
Current MLD version is 2
MLD query interval is 125 seconds
MLD querier timeout is 255 seconds
MLD max query response time is 10 seconds
Last member query response interval is 1 seconds
MLD activity: 25 joins, 17 leaves
MLD querying router is FE80::205:5FFF:FEAF:2C39 (this system)
```

The following example displays the MLD protocol messages received and sent:

```
Router# show ipv6 mld traffic

MLD Traffic Counters
Elapsed time since counters cleared:00:00:21

                              Received     Sent
Valid MLD Packets               3           1
Queries                         1           0
Reports                         2           1
Leaves                          0           0
Mtrace packets                  0           0

Errors:
Malformed Packets                           0
Bad Checksums                               0
Martian source                              0
Packets Received on MLD-disabled Interface  0
```

Example: Configuring Explicit Tracking of Receivers

```
Router> enable
Router# configure terminal
Router(config)# interface FastEthernet 1/0
Router(config-if)# ipv6 mld explicit-tracking list1
```

Example: Configuring MLD Proxy

The following example shows how to configure IPv6 MLD proxy information for the Ethernet 0/0 interface and information about the configured interface:

```
Router(config)# ipv6 mld host-proxy Ethernet0/0
Router(config)# exit
Router# show ipv6 mld host-proxy Ethernet0/0
Ethernet0/0 is up, line protocol is up
  Internet address is FE80::34/64
MLD is enabled on interface
  MLD querying router is FE80::12, Version: MLDv2
  Current MLD host version is 2
  MLD max query response time is 10 seconds
Number of MLD Query sent on interface : 10
Number of MLD Query received on interface : 20
Number of MLDv1 report sent : 5
Number of MLDv2 report sent : 10
Number of MLDv1 leave sent : 0
Number of MLDv2 leave sent : 1
```

The following example configure a group entry for the Ethernet 0/0 proxy interface and provides information about those group entries:

```
Router# show ipv6 mld host-proxy Ethernet0/0 group
Group:               FF5E::12
Uptime:             00:00:07
Group mode:     INCLUDE
Version             MLDv2
Group source list:
  Source Address        Uptime
            5000::2                     00:00:07
            2000::2                     00:01:15
Group:               FF7E::21
Uptime:             00:02:07
Group mode:     EXCLUDE
Version              MLDv2
Group source list: Empty
```

Example: Configuring PIM

The following example shows how to configure a router to use PIM-SM using 2001:DB8::1 as the RP. It sets the SPT threshold to infinity to prevent switchover to the source tree when a source starts sending traffic and sets a filter on all sources that do not have a local multicast BGP prefix.

```
Router(config)# ipv6 multicast-routing
Router(config)# ipv6 pim rp-address 2001:DB8::1
Router(config)# ipv6 pim spt-threshold infinity
Router(config)# ipv6 pim accept-register route-map reg-filter
```

Example: Configuring PIM Options

The following example sets the DR priority, the PIM hello interval, and the periodic join and prune announcement interval on Ethernet interface 0/0.

```
Router(config)# interface Ethernet0/0
Router(config)# ipv6 pim hello-interval 60
Router(config)# ipv6 pim dr-priority 3
```

Example Configuring Mroutes

The following example shows how to configure a static multicast route to be used for multicast RPF selection only:

```
Router> enable
Router# configure terminal
Router(config)# ipv6 route 2001:DB8::/64 7::7 100 multicast
```

Example Configuring an IPv6 Multiprotocol BGP Peer Group

The following example configures the IPv6 multiprotocol BGP peer group named group1:

```
router bgp 65000
no bgp default ipv4-unicast
neighbor group1 peer-group
neighbor 2001:DB8:0:CC00::1 remote-as 64600
address-family ipv6 multicast
 neighbor 3FFE:C00:0:1:A8BB:CCFF:FE00:8200 activate
 no auto-summary
 no synchronization
 exit-address-family
```

Example Redistributing Prefixes into IPv6 Multiprotocol BGP

The following example redistributes BGP routes into the IPv6 multicast database of the local router:

```
router bgp 64900
 no bgp default ipv4-unicast
address-family ipv6 multicast
 redistribute BGP
```

Example: Generating Translate Updates for IPv6 Multicast BGP

The following example shows how to generate IPv6 multicast BGP updates that correspond to unicast IPv6 updates:

```
router bgp 64900
 no bgp default ipv4-unicast
address-family ipv6 multicast
 neighbor 2001:DB8:7000::2 translate-update ipv6 multicast
```

Example: Configuring Bandwidth-Based CAC for IPv6

Example: Configuring the Interface Limit for Bandwidth-Based CAC in IPv6

The following example configures the interface limit on the source router's outgoing interface Ethernet 1/3.

```
interface Ethernet1/3
ipv6 address FE80::40:1:3 link-local
```

```
 ipv6 address 2001:DB8:1:1:3/64
 ipv6 multicast limit out acl1 10
```

Example: Configuring an Access List for Bandwidth-Based CAC in IPv6

The following example shows how to configure an access list to use for bandwidth-based CAC:

```
ipv6 access-list cost-list
   permit any ff03::1/64
```

Example: Configuring the Global Limit for Bandwidth-Based CAC

The following example configures the global limit on the source router.

```
ipv6 multicast limit cost cost-list 2
```

Example: Disabling Embedded RP Support in IPv6 PIM

The following example disables embedded RP support on IPv6 PIM:

```
Router(config)# ipv6 multicast-routing
Router(config)# no ipv6 pim rp embedded
```

Example Turning Off IPv6 PIM on a Specified Interface

The following example turns off IPv6 PIM on FastEthernet interface 1/0:

```
Router(config)# ipv6 multicast-routing
Router(config)# interface FastEthernet 1/0
Router(config)# no ipv6 pim
```

Example: Disabling MLD Router-Side Processing

The following example turns off MLD router-side processing on GigabitEthernet interface 1/0/0:

```
Router> enable
Router# configure terminal
Router(config)# interface GigabitEthernet 1/0/0

Router(config-if)# no ipv6 mld router
```

Example Disabling and Reenabling MFIB

Multicast forwarding is automatically enabled when IPv6 multicast routing is enabled; however, a user may want to disable multicast forwarding on the router. The following example shows how to disable multicast forwarding on the router and, if desired, reenable multicast forwarding on the router. The example also shows how to disable MFIB interrupt-level IPv6 multicast forwarding of outgoing packets on FastEthernet interface 1/0:

```
Router> enable
Router# configure terminal
Router(config) no ipv6 mfib
Router(config) ipv6 mfib-mode centralized-only
Router(config) interface FastEthernet 1/0
Router(config-if) no ipv6 mfib cef output
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 multicast addresses	"Implementing IPv6 Addressing and Basic Connectivity," *Cisco IOS IPv6 Configuration Guide*
Multicast BGP for IPv6	"Implementing Multiprotocol BGP for IPv6," *Cisco IOS IPv6 Configuration Guide*
IPv6 static routes	" Implementing Static Routes for IPv6," *Cisco IOS IPv6 Configuration Guide*
IPv6 tunnels	"Implementing Tunneling for IPv6," *Cisco IOS IPv6 Configuration Guide*

Standards

Title
Protocol Independent Multicast - Sparse Mode PIM-SM): Protocol Specification (Revised) , March 6, 2003
Embedding the Address of RP in IPv6 Multicast Address , May 23, 2003
PIM Upstream Detection Among Multiple Addresses , February 2003
Bi-directional Protocol Independent Multicast (BIDIR-PIM) , June 20, 2003
Bootstrap Router (BSR) Mechanism for PIM Sparse Mode , February 25, 2003

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2373	*IP Version 6 Addressing Architecture*

RFC	Title
RFC 2375	*IPv6 Multicast Address Assignments*
RFC 2460	*Internet Protocol, Version 6 (IPv6) Specification*
RFC 2461	*Neighbor Discovery for IP version 6 (IPv6)*
RFC 2462	*IPv6 Stateless Address Autoconfiguration*
RFC 2710	*Multicast Listener Discovery (MLD) for IPv6*
RFC 2711	*IPv6 Router Alert Option*
RFC 3576	*Change of Authorization*
RFC 3590	*Source Address Selection for the Multicast Listener Discovery (MLD) Protocol*
RFC 3810	*Multicast Listener Discovery Version 2 (MLDv2) for IPv6*
RFC 3956	*Embedding the Rendezvous Point (RP) Address in an IPv6 Multicast Address*
RFC 4007	*IPv6 Scoped Address Architecture*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing IPv6 Multicast

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 22 Feature Information for Implementing IPv6 Multicast

Feature Name	Releases	Feature Information
IPv6 Multicast	12.0(26)S 12.2(25)SG 12.2(33)SRA 12.2(18)SXE 12.3(2)T	IPv6 multicast allows a host to send a single data stream to a subset of all hosts simultaneously.
IPv6 Multicast: Multicast Listener Discovery (MLD) Protocol, Versions 1 and 2	12.0(26)S 12.2(18)S 12.2(25)SG 12.2(33)SRA 12.3(2)T 15.0(1)S	MLD is used by IPv6 routers to discover multicast listeners (nodes that want to receive multicast packets destined for specific multicast addresses) on directly attached links. There are two versions of MLD: MLD version 1 is based on version 2 of the IGMP for IPv4, and MLD version 2 is based on version 3 of the IGMP for IPv4. IPv6 multicast for Cisco IOS software uses both MLD version 2 and MLD version 1.
IPv6 Multicast: PIM Sparse Mode (PIM-SM)	12.0(26)S 12.2(18)S 12.2(25)SG 12.2(33)SRA 12.3(2)T 15.0(1)S	PIM-SM uses unicast routing to provide reverse-path information for multicast tree building. PIM-SM is used in a multicast network when relatively few routers are involved in each multicast and these routers do not forward multicast packets for a group, unless there is an explicit request for the traffic.
IPv6 Multicast: PIM Source Specific Multicast (PIM-SSM)	12.0(26)S 12.2(18)S 12.2(25)SG 12.2(33)SRA 12.3(2)T 15.0(1)S 15.`(1)SG	PIM-SSM supports the implementation of SSM and is derived from PIM-SM. The SSM feature forwards datagram traffic to receivers from only those multicast sources that the receivers have explicitly joined, optimizing bandwidth utilization and denying unwanted Internet broadcast traffic.

Feature Name	Releases	Feature Information
IPv6 Multicast: Scope Boundaries	12.0(26)S 12.2(18)S 12.2(25)SG 12.2(33)SRA 12.3(2)T 15.0(1)S	IPv6 includes support for global and nonglobal addresses. This section describes the usage of IPv6 addresses of different scopes.
IPv6 Multicast: MLD Access Group	12.0(26)S 12.3(4)T 12.2(25)S 12.2(25)SG 12.2(33)SRA 12.2(33)SXH 15.0(1)S	The MLD access group provides receiver access control in Cisco IOS IPv6 multicast routers.
IPv6 Multicast: PIM Accept Register	12.0(26)S 12.3(4)T 12.2(25)S 12.2(25)SG 12.2(33)SRA 12.2(33)SXH 15.0(1)S	The PIM accept register feature is the ability to perform PIM-SM register message filtering at the RP.
IPv6 Multicast: PIM Embedded RP Support	12.0(26)S 12.3(4)T 12.2(25)S 12.2(25)SG 12.2(33)SRA 12.2(33)SXH 15.0(1)S	Embedded RP support allows the router to learn RP information using the multicast group destination address instead of the statically configured RP.
IPv6 Multicast: RPF Flooding of BSR Packets	12.0(26)S 12.3(4)T 12.2(25)S 12.2(25)SG 12.2(33)SRA 12.2(33)SXH 15.0(1)S	The RPF flooding of BSR packets enables a Cisco IOS IPv6 router to not disrupt the flow of BSMs.

Feature Name	Releases	Feature Information
IPv6 Multicast: Routable Address Hello Option	12.0(26)S 12.3(4)T 12.2(25)S 12.2(25)SG 12.2(33)SRA 12.2(33)SXH 15.0(1)S	The routable address hello option adds a PIM hello message option that includes all the addresses on the interface on which the PIM hello message is advertised.
IPv6 Multicast: Static Multicast Routing (mroute)	12.0(26)S 12.3(4)T 12.2(25)S 12.2(33)SRA 12.2(33)SXH 15.0(1)S	IPv6 static mroutes share the same database as IPv6 static routes and are implemented by extending static route support.
IPv6 Multicast: Address Family Support for Multiprotocol BGP	12.0(26)S 12.2(25)S 12.2(25)SG 12.2(33)SRA 12.2(33)SXH 12.3(4)T 15.0(1)S	This feature provides multicast BGP extensions for IPv6 and supports the same features and functionality as IPv4 BGP.
IPv6 Multicast: Explicit Tracking of Receivers	12.2(25)S 12.2(25)SG 12.2(33)SRA 12.2(33)SXH 12.3(7)T 15.0(1)S	This feature allows a router to track the behavior of the hosts within its IPv6 network.
IPv6 Multicast: IPv6 Bidirectional PIM	12.2(25)SG 12.2(33)SRA 12.2(25)S 12.3(7)T 15.0(1)S	Bidirectional PIM allows multicast routers to keep reduced state information. Bidirectional shared trees convey data from sources to the RP and distribute them from the RP to the receivers.

Feature Name	Releases	Feature Information
IPv6 Multicast: MRIB	12.0(26)S 12.2(18)S 12.2(25)SG 12.3(2)T	The MRIB is a protocol-independent repository of multicast routing entries instantiated by multicast routing protocols (routing clients).
IPv6 Multicast: MFIB and MFIB Display Enhancements	12.0(26)S 12.2(18)S 12.3(2)T	The MFIB is a platform-independent and routing-protocol-independent library for IPv6 software.
IPv6 Multicast: Bootstrap Router (BSR)	12.0(28)S 12.2(25)S 12.2(25)SG 12.2(33)SRA 12.2(33)SXH 12.3(11)T 15.0(1)S	If an RP becomes unreachable, this feature allows the RP to be detected and the mapping tables modified so that the unreachable RP is no longer used, and the new tables will be rapidly distributed throughout the domain.
IPv6 Multicast: IPv6 BSR Bidirectional Support	12.2(33)SRE 12.3(14)T 15.0(1)S	Bidirectional BSR support allows bidirectional RPs to be advertised in C-RP messages and bidirectional ranges in the BSM.
IPv6 Multicast: IPv6 BSR Scoped-Zone Support	12.2(18)SXE	
IPv6 Multicast: SSM Mapping	12.2(33)SRA 12.2(18)SXE 12.4(2)T 15.0(1)S	This feature allows deployment of IPv6 SSM with hosts that are incapable of providing MLD version 2 support in their TCP/IP host stack and their IP multicast receiving application.
IPv6 Multicast: IPv6 BSR-Configure RP Mapping	12.2(33)SRE 12.2(50)SY 12.4(2)T 15.0(1)S 15.1(1)SG	This feature allows IPv6 multicast routers to be statically configured to announce scope-to-RP mappings directly from the BSR instead of learning them from candidate-RP messages.

Feature Name	Releases	Feature Information
IPv6 Multicast: MLD Group Limits	12.2(33)SRE 12.2(50)SY 12.4(2)T 15.0(1)S 15.1(1)SG	The MLD group limits feature provides protection against denial of service (DoS) attacks caused by MLD packets.
IPv6 Multicast: Multicast User Authentication and ProfileSupport	12.4(4)T	Multicast access control provides an interface between multicast and AAA for provisioning, authorizing, and accounting at the last-hop router, receiver access control functions in multicast, and group or channel disabling capability in multicast.
IPv6 Multicast: Process Switching and Fast Switching	12.0(26)S 12.2(18)S 12.3(2)T	In IPv6 multicast process switching, the Route Processor must examine, rewrite, and forward each packet. IPv6 multicast fast switching allows routers to provide better packet forwarding performance than process switching.
Distributed MFIB (dMFIB)	12.0(26)S 12.2(25)S 12.3(4)T	Distributed MFIB dMFIB is used to switch multicast IPv6 packets on distributed platforms.
IPv6: Multicast Address Group Range Support	12.2(33)SRE 12.2(33)SXI 15.0(1)M 15.0(1)S 15.1(1)SG	This feature allows the router to keep from receiving multicast traffic to be received from unauthenticated groups or unauthorized channels.
IPv6 Multicast: Bandwidth-Based Call Admission Control (CAC)	12.2(33)SRE 15.0(1)S	The bandwidth-based call admission control (CAC) for IPv6 multicast feature implements a method to monitor bandwidth per interface and multicast group avoiding oversubscription due to multicast services.
ISSU: IPv6 Multicast	15.0(1)SY	This feature is supported.
NSF/SSO: IPv6 Multicast	12.2(33)SRE 15.0(1)SY	This feature is supported in Cisco IOS Release 12.2(33)SRE.

Feature Name	Releases	Feature Information
MFIB: IPv4 SSO/ISSU	12.2(33)SRE	This feature is supported in Cisco IOS Release 12.2(33)SRE.
MLD Proxy	15.1(2)T	The MLD proxy feature enables a device to learn proxy group membership information, and forward multicast packets based upon that information.
IPv6 Multicast VRF Lite	15.1(4)M	The IPv6 Multicast VRF Lite feature provides IPv6 multicast support for multiple virtual routing/forwarding contexts (VRFs). The scope of these VRFs is limited to the router in which the VRFs are defined.

Implementing NAT-PT for IPv6

Network Address Translation--Protocol Translation (NAT-PT) is an IPv6 to IPv4 translation mechanism, as defined in RFC 2765 and RFC 2766, allowing IPv6-only devices to communicate with IPv4-only devices and vice versa.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Implementing NAT-PT for IPv6

Before implementing NAT-PT, you must configure IPv4 and IPv6 on the router interfaces that need to communicate between IPv4-only and IPv6-only networks.

Restrictions for Implementing NAT-PT for IPv6

- NAT-PT is not supported in Cisco Express Forwarding.
- NAT-PT provides limited Application Layer Gateway (ALG) support--ALG support for Internet Control Message Protocol (ICMP), File Transfer Protocol (FTP), and Domain Naming System (DNS).
- NAT-PT has the same restrictions that apply to IPv4 NAT where NAT-PT does not provide end-to-end security and the NAT-PT router can be a single point of failure in the network.

- Users must decide whether to use Static NAT-PT operation, Dynamic NAT-PT operation, Port Address Translation (PAT), or IPv4-mapped operation. Deciding which operation to use determines how a user will configure and operate NAT-PT.
- Bridge-Group Virtual interfaces (BVIs) in IPv6 are not supported with NAT-PT and wireless interfaces Dot11Radio.

Information About Implementing NAT-PT for IPv6

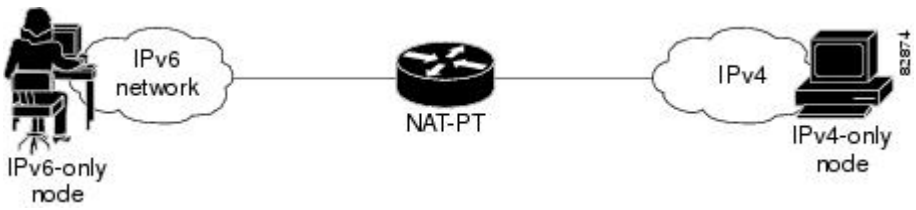
Users can configure NAT-PT using one of the following operations--static NAT-PT, dynamic NAT-PT, Port Address Translation (PAT), or IPv4-mapped operation--which are described in the following sections:

NAT-PT Overview

NAT-PT for Cisco software was designed using RFC 2766 and RFC 2765 as a migration tool to help customers transition their IPv4 networks to IPv6 networks. Using a protocol translator between IPv6 and IPv4 allows direct communication between hosts speaking a different network protocol. Users can use either static definitions or IPv4-mapped definitions for NAT-PT operation.

The figure below shows that NAT-PT runs on a router between an IPv6 network and an IPv4 network to connect an IPv6-only node with an IPv4-only node.

Figure 37 NAT-PT Basic Operation



Although IPv6 solves addressing issues for customers, a long transition period is likely before customers move to an exclusive IPv6 network environment. During the transition period, any new IPv6-only networks will need to continue to communicate with existing IPv4 networks. NAT-PT is designed to be deployed to allow direct communication between IPv6-only networks and IPv4-only networks. For a service provider customer, an example could be an IPv6-only client trying to access an IPv4-only web server. Enterprise customers will also migrate to IPv6 in stages, and many of their IPv4-only networks will be operational for several years. Dual-stack networks may have some IPv6-only hosts configured to take advantage of the IPv6 autoconfiguration, global addressing, and simpler management, and these hosts can use NAT-PT to communicate with existing IPv4-only networks in the same organization.

One of the benefits of NAT-PT is that no changes are required to existing hosts, because all the NAT-PT configurations are performed at the NAT-PT router. Customers with existing stable IPv4 networks can introduce an IPv6 network and use NAT-PT to allow communication without disrupting the existing network. To further illustrate the seamless transition, File Transfer Protocol (FTP) can be used between IPv4 and IPv6 networks, just as within an IPv4 network. Packet fragmentation is enabled by default when

IPv6 is configured, allowing IPv6 and IPv4 networks to resolve fragmentation problems between the networks. Without the ability to resolve fragmentation, connectivity could become intermittent when fragmented packets might be dropped or improperly interpreted.

Cisco has developed other transition techniques including dual stack, IPv6 over MPLS, and tunneling. NAT-PT should not be used when other native communication techniques exist. If a host is configured as a dual-stack host with both IPv4 and IPv6, we do not recommend using NAT-PT to communicate between the dual-stack host and an IPv6-only or IPv4-only host. NAT-PT is not recommended for a scenario in which an IPv6-only network is trying to communicate to another IPv6-only network via an IPv4 backbone or vice versa, because NAT-PT would require a double translation to be performed. In this scenario, tunneling techniques are recommended.

The following sections describe the operations that may be used to configure NAT-PT. Users have the option to use one of the following operations for NAT-PT operation, but not all four.

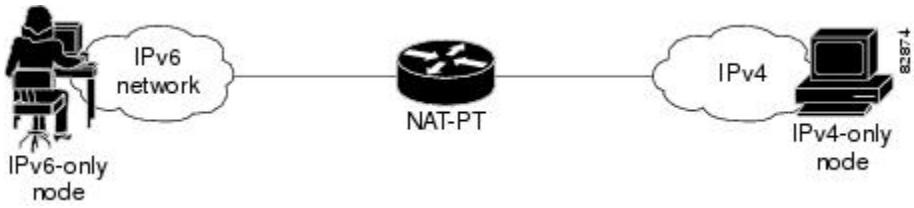
Static NAT-PT Operation

Static NAT-PT uses static translation rules to map one IPv6 address to one IPv4 address. IPv6 network nodes communicate with IPv4 network nodes using an IPv6 mapping of the IPv4 address configured on the NAT-PT router.

The figure below shows how the IPv6-only node named A can communicate with the IPv4-only node named C using NAT-PT. The NAT-PT device is configured to map the source IPv6 address for node A of 2001:DB8:bbbb:1::1 to the IPv4 address 192.168.99.2. NAT-PT is also configured to map the source address of IPv4 node C, 192.168.30.1 to 2001:DB8::a. When packets with a source IPv6 address of node A are received at the NAT-PT router, they are translated to have a destination address to match node C in the IPv4-only network. NAT-PT can also be configured to match a source IPv4 address and translate the packet to an IPv6 destination address to allow an IPv4-only host communicate with an IPv6-only host.

If you have multiple IPv6-only or IPv4-only hosts that need to communicate, you may need to configure many static NAT-PT mappings. Static NAT-PT is useful when applications or servers require access to a stable IPv4 address, such as accessing an external IPv4 DNS server.

Figure 38 Static NAT-PT Operation



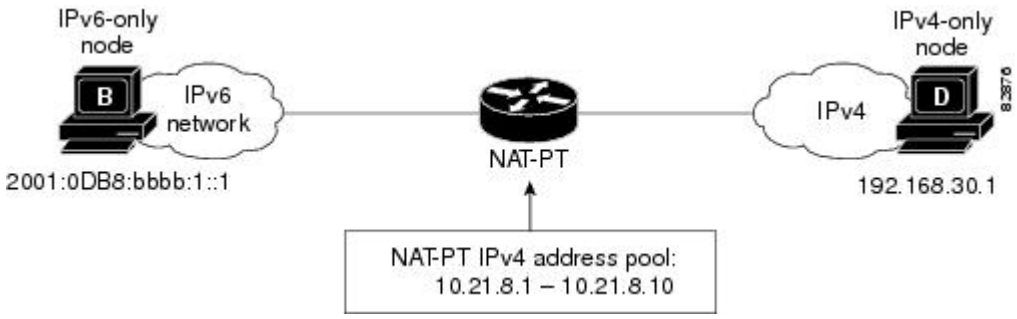
Dynamic NAT-PT Operation

Dynamic NAT-PT allows multiple NAT-PT mappings by allocating addresses from a pool. NAT-PT is configured with a pool of IPv6 and/or IPv4 addresses. At the start of a NAT-PT session a temporary address is dynamically allocated from the pool. The number of addresses available in the address pool determines the maximum number of concurrent sessions. The NAT-PT device records each mapping between addresses in a dynamic state table.

The figure below shows how dynamic NAT-PT operates. The IPv6-only node B can communicate with the IPv4-only node D using dynamic NAT-PT. The NAT-PT device is configured with an IPv6 access list, prefix list, or route map to determine which packets are to be translated by NAT-PT. A pool of IPv4 addresses--10.21.8.1 to 10.21.8.10 in the figure -- is also configured. When an IPv6 packet to be translated

is identified, NAT-PT uses the configured mapping rules and assigns a temporary IPv4 address from the configured pool of IPv4 addresses.

Figure 39 Dynamic NAT-PT Operation



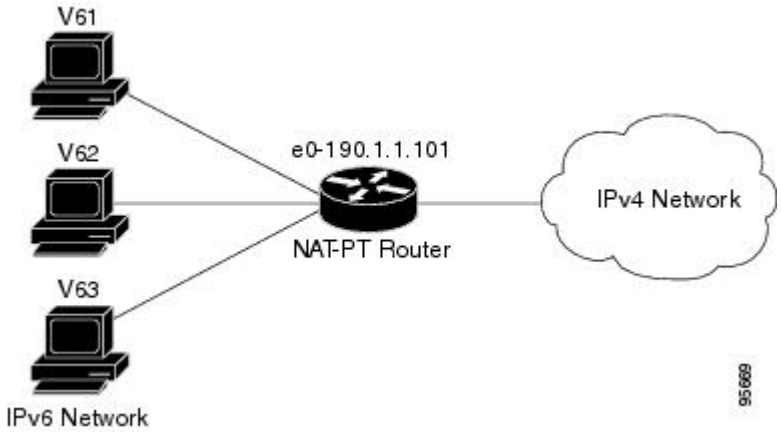
Dynamic NAT-PT translation operation requires at least one static mapping for the IPv4 DNS server.

After the IPv6 to IPv4 connection is established, the reply packets going from IPv4 to IPv6 take advantage of the previously established dynamic mapping to translate back from IPv4 to IPv6. If the connection is initiated by an IPv4-only host, then the explanation is reversed.

Port Address Translation or Overload

Port Address Translation (PA T), also known as Overload, allows a single IPv4 address to be used among multiple sessions by multiplexing on the port number to associate several IPv6 users with a single IPv4 address. PAT can be accomplished through a specific interface or through a pool of addresses. The figure below shows multiple IPv6 addresses from the IPv6 network linked to a single IPv4 interface into the IPv4 network.

Figure 40 Port Address Translation



IPv4-Mapped Operation

Customers can also send traffic from their IPv6 network to an IPv4 network without configuring IPv6 destination address mapping. A packet arriving at an interface is checked to discover if it has a NAT-PT prefix that was configured with the **ipv6 nat prefix v4-mapped** command. If the prefix matches, then an

access-list check is performed to discover if the source address matches the access list or prefix list. If the prefix does not match, the packet is dropped.

If the prefix matches, source address translation is performed. If a rule has been configured for the source address translation, the last 32 bits of the destination IPv6 address is used as the IPv4 destination and a flow entry is created.

With an IPv4-mapping configuration on the router, when the DNS ALG IPv4 address is converted to an IPv6 address, the IPv6 address is processed and the DNS packets from IPv4 network get their ALGs translated into the IPv6 network.

How to Implement NAT-PT for IPv6

Configuring Basic IPv6 to IPv4 Connectivity for NAT-PT for IPv6

Perform this task to configure basic IPv6 to IPv4 connectivity for NAT-PT, which consists of configuring the NAT-PT prefix globally, and enable NAT-PT on an interface. For NAT-PT to be operational, NAT-PT must be enabled on both the incoming and outgoing interfaces.

An IPv6 prefix with a prefix length of 96 must be specified for NAT-PT to use. The IPv6 prefix can be a unique local unicast prefix, a subnet of your allocated IPv6 prefix, or even an extra prefix obtained from your Internet service provider (ISP). The NAT-PT prefix is used to match a destination address of an IPv6 packet. If the match is successful, NAT-PT will use the configured address mapping rules to translate the IPv6 packet to an IPv4 packet. The NAT-PT prefix can be configured globally or with different IPv6 prefixes on individual interfaces. Using a different NAT-PT prefix on several interfaces allows the NAT-PT router to support an IPv6 network with multiple exit points to IPv4 networks.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 nat prefix** *ipv6-prefix* / *prefix-length*
4. **interface** *type number*
5. **ipv6 address** *ipv6-address* {/*prefix-length* | **link-local**}
6. **ipv6 nat**
7. **exit**
8. **interface** *type number*
9. **ip address** *ip-address mask* [**secondary**]
10. **ipv6 nat**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 nat prefix** *ipv6-prefix* / *prefix-length* **Example:** `Router# ipv6 nat prefix 2001:DB8::/96`	Assigns an IPv6 prefix as a global NAT-PT prefix. • Matching destination prefixes in IPv6 packets are translated by NAT-PT. • The only prefix length supported is 96.
Step 4	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 3/1/1`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 5	**ipv6 address** *ipv6-address* {/*prefix-length* | **link-local**} **Example:** `Router(config-if)# ipv6 address 2001:DB8:yyyy:1::9/64`	Specifies an IPv6 address assigned to the interface and enables IPv6 processing on the interface.
Step 6	**ipv6 nat** **Example:** `Router(config-if)# ipv6 nat`	Enables NAT-PT on the interface.
Step 7	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode, and returns the router to global configuration mode.

	Command or Action	Purpose
Step 8	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 3/3/1`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 9	**ip address** *ip-address mask* [**secondary**] **Example:** `Router(config-if)# ip address 192.168.30.9 255.255.255.0`	Specifies an IP address and mask assigned to the interface and enables IP processing on the interface.
Step 10	**ipv6 nat** **Example:** `Router(config-if)# ipv6 nat`	Enables NAT-PT on the interface.

Configuring IPv4-Mapped NAT-PT

Perform this task to enable customers to send traffic from their IPv6 network to an IPv4 network without configuring IPv6 destination address mapping. This task shows the **ipv6 nat prefix v4-mapped** command configured on a specified interface, but the command could alternatively be configured globally:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 nat prefix** *ipv6-prefix* **v4-mapped** {*access-list-name* | *ipv6-prefix*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 3/1/1`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**ipv6 nat prefix** *ipv6-prefix* **v4-mapped** {*access-list-name* | *ipv6-prefix*} **Example:** `Router(config-if)# ipv6 nat prefix 2001::/96 v4-mapped v4mapacl`	Enables customers to send traffic from their IPv6 network to an IPv4 network without configuring IPv6 destination address mapping.

Configuring Mappings for IPv6 Hosts Accessing IPv4 Hosts

Perform this task to configure static or dynamic IPv6 to IPv4 address mappings. The dynamic address mappings include assigning a pool of IPv4 addresses and using an access list, prefix list, or route map to define which packets are to be translated.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. Do one of the following:
   - **ipv6 nat v6v4 source** *ipv6-address ipv4-address*
   -
   - **ipv6 nat v6v4 source** {**list** *access-list-name* | **route-map** *map-name*} **pool** *name*
4. **ipv6 nat v6v4 pool** *name start-ipv4 end-ipv4* **prefix-length** *prefix-length*
5. **ipv6 nat translation** [**max-entries** *number*] {**timeout** | **udp-timeout** | **dns-timeout**| **tcp-timeout**| **finrst-timeout**| **icmp-timeout**} {*seconds*| **never**}
6. **ipv6 access-list** *access-list-name*
7. **permit** *protocol* {*source-ipv6-prefix / prefix-length*| **any**| **host** *source-ipv6-address*}[*operator* [*port-number*]] {*destination-ipv6-prefix / prefix-length*| **any**| **host** *destination-ipv6-address*}
8. **exit**
9. **show ipv6 nat translations** [**icmp**| **tcp**| **udp**] [**verbose**]
10. **show ipv6 nat statistics**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

Step 3	Do one of the following:
	• **ipv6 nat v6v4 source** *ipv6-address ipv4-address*
	•
	• **ipv6 nat v6v4 source** {**list** *access-list-name* | **route-map** *map-name*} **pool** *name*
	
	Example:
	`Router(config)# ipv6 nat v6v4 source 2001:DB8:yyyy:1::1 10.21.8.10`
	
	Example:
	
	Example:
	`Router(config)# ipv6 nat v6v4 source list pt-list1 pool v4pool`
		Enables a static IPv6 to IPv4 address mapping using NAT-PT.
		or
		Enables a dynamic IPv6 to IPv4 address mapping using NAT-PT.
		• Use the **list** or **route-map** keyword to specify a prefix list, access list, or a route map to define which packets are translated.
		• Use the **pool** keyword to specify the name of a pool of addresses, created by the **ipv6 nat v6v4 pool**command, to be used in dynamic NAT-PT address mapping.

Step 4	**ipv6 nat v6v4 pool** *name start-ipv4 end-ipv4* **prefix-length** *prefix-length* **Example:** `Router(config)# ipv6 nat v6v4 pool v4pool 10.21.8.1 10.21.8.10 prefix-length 24`	Specifies a pool of IPv4 addresses to be used by NAT-PT for dynamic address mapping.

	Command or Action	Purpose
Step 5	**ipv6 nat translation** [**max-entries** *number*] {**timeout** | **udp-timeout** | **dns-timeout**| **tcp-timeout**| **finrst-timeout**| **icmp-timeout**} {*seconds*| **never**} **Example:** `Router(config)# ipv6 nat translation udp-timeout 600`	(Optional) Specifies the time after which NAT-PT translations time out.
Step 6	**ipv6 access-list** *access-list-name* **Example:** `Router(config)# ipv6 access-list pt-list1`	(Optional) Defines an IPv6 access list and enters IPv6 access list configuration mode. The router prompt changes to Router(config-ipv6-acl)#. • The *access-list name* argument specifies the name of the IPv6 access control list (ACL). IPv6 ACL names cannot contain a space or quotation mark, or begin with a numeral.
Step 7	**permit** *protocol* {*source-ipv6-prefix* / *prefix-length*| **any**| **host** *source-ipv6-address*}[*operator* [*port-number*]] {*destination-ipv6-prefix* / *prefix-length*| **any**| **host** *destination-ipv6-address*} **Example:** `Router(config-ipv6-acl)# permit ipv6 2001:DB8:bbbb:1::/64 any`	(Optional) Specifies permit conditions for an IPv6 ACL. • The *protocol* argument specifies the name or number of an Internet protocol. It can be one of the keywords **ahp**, **esp**, **icmp**, **ipv6**, **pcp**, **sctp**, **tcp**, or **udp**, or an integer in the range from 0 to 255 representing an IPv6 protocol number. • The *source-ipv6-prefix* / *prefix-length* and *destination-ipv6-prefix* / *prefix-length* arguments specify the source and destination IPv6 network or class of networks about which to set permit conditions. These arguments must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons. • The **any** keyword is an abbreviation for the IPv6 prefix ::/0. • The **host** *source-ipv6-address* keyword and argument combination specifies the source IPv6 host address about which to set permit conditions. The *source-ipv6-address*argument must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons. • Only the arguments and keywords relevant to this task are specified here. Refer to the **permit** command in the *IPv6 for Cisco IOS Command Reference* document for information on supported arguments and keywords.
Step 8	**exit** **Example:** `Router(config-if)# exit`	Exits access list configuration mode, and returns the router to global configuration mode. Enter the **exit** command twice to return to privileged EXEC mode.

	Command or Action	Purpose
Step 9	**show ipv6 nat translations** [**icmp**| **tcp**| **udp**] [**verbose**] **Example:** `Router# show ipv6 nat translations verbose`	(Optional) Displays active NAT-PT translations. • Use the optional **icmp**, **tcp**, and **udp** keywords to display detailed information about the NAT-PT translation events for the specified protocol. • Use the optional **verbose** keyword to display more detailed information about the active translations.
Step 10	**show ipv6 nat statistics** **Example:** `Router# show ipv6 nat statistics`	(Optional) Displays NAT-PT statistics.

Configuring Mappings for IPv4 Hosts Accessing IPv6 Hosts

Perform this optional task to configure static or dynamic IPv4 to IPv6 address mappings. The dynamic address mappings include assigning a pool of IPv6 addresses and using an access list, prefix list, or route map to define which packets are to be translated.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. Do one of the following:
   - **ipv6 nat v4v6 source** *ipv6-address ipv4-address*
   -
   - **ipv6 nat v4v6 source list** {*access-list-number* | *name*} **pool** *name*
4. **ipv6 nat v4v6 pool** *name start-ipv6 end-ipv6* **prefix-length** *prefix-length*
5. **access-list** {*access-list-name*| *number*}{**deny**| **permit**} [*source source-wildcard*] [**log**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

Step 3	Do one of the following:
	• **ipv6 nat v4v6 source** *ipv6-address ipv4-address*
	•
	• **ipv6 nat v4v6 source list** {*access-list-number* | *name*} **pool** *name*
	
	Example:
	`Router(config)# ipv6 nat v4v6 source 10.21.8.11 2001:DB8:yyyy::2`
	
	Example:
	
	Example:
	`Router(config)# ipv6 nat v4v6 source list 1 pool v6pool`
		Enables a static IPv4 to IPv6 address mapping using NAT-PT.
		or
		Enables a dynamic IPv4 to IPv6 address mapping using NAT-PT.
		• Use the **list** keyword to specify an access list to define which packets are translated.
		• Use the **pool** keyword to specify the name of a pool of addresses, created by the **ipv6 nat v4v6 pool**command, to be used in dynamic NAT-PT address mapping.

Step 4	**ipv6 nat v4v6 pool** *name start-ipv6 end-ipv6* **prefix-length** *prefix-length* **Example:** `Router(config)# ipv6 nat v4v6 pool v6pool 2001:DB8:yyyy::1 2001:DB8:yyyy::2 prefix-length 128`	Specifies a pool of IPv6 addresses to be used by NAT-PT for dynamic address mapping.
Step 5	**access-list** {*access-list-name*| *number*}{**deny**| **permit**} [*source source-wildcard*] [**log**] **Example:** `Router(config)# access-list 1 permit 192.168.30.0 0.0.0.255`	Specifies an entry in a standard IPv4 access list.

Configuring PAT for IPv6 to IPv4 Address Mappings

Perform this task to configure PAT for IPv6 to IPv4 address mappings. Multiple IPv6 addresses are mapped to a single IPv4 address or to a pool of IPv4 addresses and using an access list, prefix list, or route map to define which packets are to be translated.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. Do one of the following:
   - **ipv6 nat v6v4 source** {**list** *access-list-name* | **route-map** *map-name*} **pool** *name overload*
   -
   - **ipv6 nat v6v4 source** {**list** *access-list-name* | **route-map** *map-name*} **interface** *interface name* **overload**
4. **ipv6 nat v6v4 pool** *name start-ipv4 end-ipv4* **prefix-length** *prefix-length*
5. **ipv6 nat translation** [**max-entries** *number*] {**timeout** | **udp-timeout** | **dns-timeout**| **tcp-timeout**| **finrst-timeout**| **icmp-timeout**} {*seconds*| **never**}
6. **ipv6 access-list** *access-list-name*
7. **permit** *protocol* {*source-ipv6-prefix / prefix-length*| **any**| **host** *source-ipv6-address*}[*operator* [*port-number*]] {*destination-ipv6-prefix / prefix-length*| **any**| **host** *destination-ipv6-address*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

Command or Action	Purpose

Step 3 Do one of the following:
• **ipv6 nat v6v4 source** {**list** *access-list-name* | **route-map** *map-name*} **pool** *name overload*
•
• **ipv6 nat v6v4 source** {**list** *access-list-name* | **route-map** *map-name*} **interface** *interface name* **overload**
Example:
`Router(config)# ipv6 nat v6v4 source 2001:DB8:yyyy:1::1 10.21.8.10`
Example:
Example:
`Router(config)# ipv6 nat v6v4 source list pt-list1 pool v4pool overload`
	Enables a dynamic IPv6 to IPv4 address overload mapping using a pool address.
	or
	Enables a dynamic IPv6 to IPv4 address overload mapping using an interface address.
	• Use the **list** or **route-map** keyword to specify a prefix list, access list, or a route map to define which packets are translated.
	• Use the **pool** keyword to specify the name of a pool of addresses, created by the **ipv6 nat v6v4 pool**command, to be used in dynamic NAT-PT address mapping.
	• Use the interface keyword to specify the interface address to be used for overload.

Step 4 **ipv6 nat v6v4 pool** *name start-ipv4 end-ipv4* **prefix-length** *prefix-length* **Example:** `Router(config)# ipv6 nat v6v4 pool v4pool 10.21.8.1 10.21.8.10 prefix-length 24`	Specifies a pool of IPv4 addresses to be used by NAT-PT for dynamic address mapping.
Step 5 **ipv6 nat translation** [**max-entries** *number*] {**timeout** | **udp-timeout** | **dns-timeout**| **tcp-timeout**| **finrst-timeout**| **icmp-timeout**} {*seconds*| **never**} **Example:** `Router(config)# ipv6 nat translation udp-timeout 600`	(Optional) Specifies the time after which NAT-PT translations time out.

	Command or Action	Purpose
Step 6	**ipv6 access-list** *access-list-name* **Example:** `Router(config)# ipv6 access-list pt-list1`	(Optional) Defines an IPv6 access list and enters IPv6 access list configuration mode. The router prompt changes to Router(config-ipv6-acl)#. • The *access-list name* argument specifies the name of the IPv6 access control list (ACL). IPv6 ACL names cannot contain a space or quotation mark, or begin with a numeral.
Step 7	**permit** *protocol* {*source-ipv6-prefix* / *prefix-length*| **any**| **host** *source-ipv6-address*}[*operator* [*port-number*]] {*destination-ipv6-prefix* / *prefix-length*| **any**| **host** *destination-ipv6-address*} **Example:** `Router(config-ipv6-acl)# permit ipv6 2001:DB8:bbbb:1::/64 any`	(Optional) Specifies permit conditions for an IPv6 ACL. • The *protocol* argument specifies the name or number of an Internet protocol. It can be one of the keywords **ahp**, **esp**, **icmp**, **ipv6**, **pcp**, **sctp**, **tcp**, or **udp**, or an integer in the range from 0 to 255 representing an IPv6 protocol number. • The *source-ipv6-prefix* / *prefix-length* and *destination-ipv6-prefix* / *prefix-length* arguments specify the source and destination IPv6 network or class of networks about which to set permit conditions. These arguments must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons. • The **any** keyword is an abbreviation for the IPv6 prefix ::/0. • The **host** *source-ipv6-address* keyword and argument combination specifies the source IPv6 host address about which to set permit conditions. The *source-ipv6-address*argument must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons.

Verifying NAT-PT Configuration and Operation

SUMMARY STEPS

1. **clear ipv6 nat translation ***
2. **enable**
3. **debug ipv6 nat** [**detailed**| **port**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**clear ipv6 nat translation *** **Example:** `Router> clear ipv6 nat translation *`	(Optional) Clears dynamic NAT-PT translations from the dynamic translation state table. • Use the ***** keyword to clear all dynamic NAT-PT translations. **Note** Static translation configuration is not affected by this command.

	Command or Action	Purpose
Step 2	**enable** **Example:** `Router> enable`	Enables higher privilege levels, such as privileged EXEC mode. • Enter your password if prompted.
Step 3	**debug ipv6 nat** [**detailed**| **port**] **Example:** `Router# debug ipv6 nat detail`	Displays debugging messages for NAT-PT translation events.

• Examples, page 570

Examples

Sample Output from the show ipv6 nat translations Command

In the following example, output information about active NAT-PT translations is displayed using the **show ipv6 nat translations**command:

```
Router# show ipv6 nat translations
Prot  IPv4 source              IPv6 source
      IPv4 destination         IPv6 destination
---   ---                      ---
      192.168.123.2            2001:DB8::2
---   ---                      ---
      192.168.122.10           2001:DB8::10
tcp   192.168.124.8,11047      2001:DB8:3::8,11047
      192.168.123.2,23         2001:DB8::2,23
udp   192.168.124.8,52922      2001:DB8:3::8,52922
      192.168.123.2,69         2001::2,69
udp   192.168.124.8,52922      2001:DB8:3::8,52922
      192.168.123.2,52922      2001:DB8::2,52922
---   192.168.124.8            2001:DB8:3::8
      192.168.123.2            2001:DB8::2
---   192.168.124.8            2001:DB8:3::8
      ---                      ---
---   192.168.121.4            2001:DB8:5::4
      ---                      ---
```

In the following example, detailed output information about active NAT-PT translations is displayed using the **show ipv6 nat translations**command with the **verbose** keyword:

```
Router# show ipv6 nat translations verbose
Prot  IPv4 source              IPv6 source
      IPv4 destination         IPv6 destination
---   ---                      ---
      192.168.123.2            2001:DB8::2
      create 00:04:24, use 00:03:24,
---   ---                      ---
      192.168.122.10           2001:DB8::10
      create 00:04:24, use 00:04:24,
tcp   192.168.124.8,11047      2001:DB8:3::8,11047
      192.168.123.2,23         2001:DB8::2,23
      create 00:03:24, use 00:03:20, left 00:16:39,
```

```
udp   192.168.124.8,52922      2001:DB8:3::8,52922
      192.168.123.2,69         2001:DB8::2,69
      create 00:02:51, use 00:02:37, left 00:17:22,
udp   192.168.124.8,52922      2001:DB8:3::8,52922
      192.168.123.2,52922      2001:DB8::2,52922
      create 00:02:48, use 00:02:30, left 00:17:29,
---   192.168.124.8            2001:DB8:3::8
      192.168.123.2            2001:DB8::2
      create 00:03:24, use 00:02:34, left 00:17:25,
---   192.168.124.8            2001:DB8:3::8
      ---                      ---
      create 00:04:24, use 00:03:24,
---   192.168.121.4            2001:DB8:5::4
      ---                      ---
      create 00:04:25, use 00:04:25,
```

Sample Output from the show ipv6 nat statistics Command

In the following example, output information about NAT-PT statistics is displayed using the **show ipv6 nat statistics**command:

```
Router# show ipv6 nat statistics
Total active translations: 4 (4 static, 0 dynamic; 0 extended)
NAT-PT interfaces:
  Ethernet3/1, Ethernet3/3
Hits: 0  Misses: 0
Expired translations: 0
```

Sample Output from the clear ipv6 nat translation Command

In the following example, all dynamic NAT-PT translations are cleared from the dynamic translation state table using the **clear ipv6 nat translation**command with the ***** keyword. When the output information about active NAT-PT translations is then displayed using the **show ipv6 nat translations**command, only the static translation configurations remain. Compare this **show** command output with the output from the **show ipv6 nat translations**command in Step 1.

```
Router# clear ipv6 nat translation *
Router# show ipv6 nat translations
Prot  IPv4 source              IPv6 source
      IPv4 destination         IPv6 destination
---   ---                      ---
      192.168.123.2            2001:DB8::2
---   ---                      ---
      192.168.122.10           2001:DB8::10
---   192.168.124.8            2001:DB8:3::8
      ---                      ---
---   192.168.121.4            2001:DB8:5::4
      ---                      ---
```

Sample Output from the debug ipv6 nat Command

In the following example, debugging messages for NAT-PT translation events are displayed using the **debug ipv6 nat**command:

```
Router# debug ipv6 nat
00:06:06: IPv6 NAT: icmp src (2001:DB8:3002::8) -> (192.168.124.8), dst
(2001:DB8:2001::2) -> (192.168.123.2)
00:06:06: IPv6 NAT: icmp src (192.168.123.2) -> (2001:DB8:2001::2), dst (192.168.124.8) -
> (2001:DB8:3002::8)
00:06:06: IPv6 NAT: icmp src (2001:DB8:3002::8) -> (192.168.124.8), dst
(2001:DB8:2001::2) -> (192.168.123.2)
00:06:06: IPv6 NAT: icmp src (192.168.123.2) -> (2001:DB8:2001::2), dst (192.168.124.8) -
> (2001:DB8:3002::8)
00:06:06: IPv6 NAT: tcp src (2001:DB8:3002::8) -> (192.168.124.8), dst (2001:DB8:2001::2)
-> (192.168.123.2)
00:06:06: IPv6 NAT: tcp src (192.168.123.2) -> (2001:DB8:2001::2), dst (192.168.124.8) ->
```

```
(2001:DB8:3002::8)
00:06:06: IPv6 NAT: tcp src (2001:DB8:3002::8) -> (192.168.124.8), dst (2001:DB8:2001::2)
-> (192.168.123.2)
00:06:06: IPv6 NAT: tcp src (2001:DB8:3002::8) -> (192.168.124.8), dst (2001:DB8:2001::2)
-> (192.168.123.2)
00:06:06: IPv6 NAT: tcp src (2001:DB8:3002::8) -> (192.168.124.8), dst (2001:DB8:2001::2)
-> (192.168.123.2)
00:06:06: IPv6 NAT: tcp src (192.168.123.2) -> (2001:DB8:2001::2), dst (192.168.124.8) ->
(2001:DB8:3002::8)
```

Configuration Examples for NAT-PT for IPv6

Example: Static NAT-PT Configuration

The following example configures the NAT-PT prefix globally, enables NAT-PT on two interfaces, and configures two static NAT-PT mappings. Ethernet interface 3/1 is configured as IPv6 only, and Ethernet interface 3/3 is configured as IPv4 only.

```
interface Ethernet3/1
 ipv6 address 2001:DB8:3002::9/64
 ipv6 enable
 ipv6 nat
!
interface Ethernet3/3
 ip address 192.168.30.9 255.255.255.0
 ipv6 nat
!
ipv6 nat v4v6 source 192.168.30.1 2001:DB8:0::2
ipv6 nat v6v4 source 2001:DB8:bbbb:1::1 10.21.8.10
ipv6 nat prefix 2001:DB8:0::/96
```

Example: Enabling Traffic to be Sent from an IPv6 Network to an IPv4 Network

In the following example, the access list permits any IPv6 source address with the prefix 2001::/96 to go to the destination with a 2000::/96 prefix. The destination is then translated to the last 32 bit of its IPv6 address; for example: source address = 2001::1, destination address = 2000::192.168.1.1. The destination then becomes 192.168.1.1 in the IPv4 network:

```
ipv6 nat prefix 2000::/96 v4-mapped v4map_acl
ipv6 access-list v4map_acl
 permit ipv6 2001::/96 2000::/96
```

Example: Dynamic NAT-PT Configuration for IPv6 Hosts Accessing IPv4 Hosts

The following example configures the NAT-PT prefix globally, enables NAT-PT on two interfaces, and configures one static NAT-PT mapping (used, for example, to access a DNS server). A dynamic NAT-PT mapping is also configured to map IPv6 addresses to IPv4 addresses using a pool of IPv4 addresses named

v4pool. The packets to be translated by NAT-PT are filtered using an IPv6 access list named pt-list1. The User Datagram Protocol (UDP) translation entries are configured to time out after 10 minutes. Ethernet interface 3/1 is configured as IPv6 only, and Ethernet interface 3/3 is configured as IPv4 only.

```
interface Ethernet3/1
 ipv6 address 2001:DB8:bbbb:1::9/64
 ipv6 enable
 ipv6 nat
!
interface Ethernet3/3
 ip address 192.168.30.9 255.255.255.0
 ipv6 nat
!
ipv6 nat v4v6 source 192.168.30.1 2001:DB8:0::2
ipv6 nat v6v4 source list pt-list1 pool v4pool
ipv6 nat v6v4 pool v4pool 10.21.8.1 10.21.8.10 prefix-length 24
ipv6 nat translation udp-timeout 600
ipv6 nat prefix 2001:DB8:1::/96
!
ipv6 access-list pt-list1
 permit ipv6 2001:DB8:bbbb:1::/64 any
```

Example: Dynamic NAT-PT Configuration for IPv4 Hosts Accessing IPv6 Hosts

The following example configures the NAT-PT prefix globally, enables NAT-PT on two interfaces, and configures one static NAT-PT mapping (used, for example, to access a DNS server). A dynamic NAT-PT mapping is also configured to map IPv4 addresses to IPv6 addresses using a pool of IPv6 addresses named v6pool. The packets to be translated by NAT-PT are filtered using an access list named pt-list2. Ethernet interface 3/1 is configured as IPv6 only, and Ethernet interface 3/3 is configured as IPv4 only.

```
interface Ethernet3/1
 ipv6 address 2001:DB8:bbbb:1::9/64
 ipv6 enable
 ipv6 nat
!
interface Ethernet3/3
 ip address 192.168.30.9 255.255.255.0
 ipv6 nat
!
ipv6 nat v4v6 source list 72 pool v6pool
ipv6 nat v4v6 pool v6pool 2001:DB8:0::1 2001:DB8:0::2 prefix-length 128
ipv6 nat v6v4 source 2001:DB8:bbbb:1::1 10.21.8.0
ipv6 nat prefix 2001:DB8:0::/96
!
access-list 72 permit 192.168.30.0 0.0.0.255
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 supported feature list	"Start Here: Cisco IOS Software Release Specifics for IPv6 Features," *Cisco IOS IPv6 Configuration Guide*

Related Topic	Document Title
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*
IPv6 addressing and IPv6 addressing services	Implementing IPv6 Addressing and Basic Connectivity
IPv4 addressing services commands	*Cisco IOS IP Addressing Services Command Reference*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIBs	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 2765	*Stateless IP/ICMP Translation Algorithm (SIIT)*
RFC 2766	*Network Address Translation - Protocol Translation (NAT-PT)*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing NAT-PT for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 23 Feature Information for Implementing NAT-PT for IPv6

Feature Name	Releases	Feature Information
NAT Protocol Translation	12.2(13)T 12.3 12.3(2)T 12.4	NAT-PT is an IPv6-IPv4 translation mechanism that allows IPv6-only devices to communicate with IPv4-only devices and vice versa. NAT-PT is not supported in Cisco Express Forwarding.
NAT-PT--Support for DNS ALG	12.2(13)T 12.3 12.3(2)T 12.4	IPv6 provides DNS ALG support.
NAT-PT--Support for FTP ALG	12.3(2)T 12.4 12.4(2)T	IPv6 provides FTP ALG support.
NAT-PT--Support for Fragmentation	12.3(2)T 12.4 12.4(2)T	Packet fragmentation is enabled by default when IPv6 is configured, allowing IPv6 and IPv4 networks to resolve fragmentation problems between the networks.

Feature Name	Releases	Feature Information
NAT-PT--Support for Overload (PAT)	12.3(2)T 12.4 12.4(2)T	PAT, also known as Overload, allows a single IPv4 address to be used among multiple sessions by multiplexing on the port number to associate several IPv6 users with a single IPv4 address.

Netflow v9 for IPv6

This module contains information about and instructions for configuring NetFlow and NetFlow Data Export (NDE) for capturing and exporting data from IP version 6 (IPv6) traffic flows using the NetFlow version 9 (v9) export format.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Netflow v9 for IPv6

Your router must be running Cisco IOS release 12.2(33)SRB or later to configure the Netflow v9 for IPv6 feature.

Information About Netflow v9 for IPv6

NetFlow and NDE on the PFC

The NetFlow cache on the PFC captures statistics for flows routed in hardware.

The PFC uses one of these flow masks to create NetFlow entries:

- **source-only** --The cache contains one entry for each source IP address. All flows from a given source IP address use this entry.
- **destination** --The cache contains one entry for each destination IP address. All flows to a given destination IP address use this entry.
- **destination-source** --The cache contains one entry for each source and destination IP address pair. All flows between the same source and destination IP addresses use this entry.
- **destination-source-interface** --Adds the source VLAN SNMP ifIndex to the information in the **destination-source** flow mask.
- **full** --A separate cache entry is created for each IP flow. A full entry includes the source IP address, destination IP address, protocol, and protocol interfaces.
- **full-interface** --Adds the source VLAN SNMP ifIndex to the information in the **full** flow mask.

NetFlow Export Format Version 9

For all NetFlow export versions, the NetFlow export datagram consists of a header and a sequence of flow records. The header contains information such as sequence number, record count, and system uptime. The flow record contains flow information, such as IP addresses, ports, and routing information.

NetFlow version 9 export format is the newest NetFlow export format. The distinguishing feature of the NetFlow version 9 export format is that it is template based. Templates make the record format extensible. NetFlow version 9 export format allows future enhancements to NetFlow without requiring concurrent changes to the basic flow-record format.

The NetFlow version 9 export record format is different from the traditional NetFlow fixed format export record. In NetFlow version 9, a template describes the NetFlow data, and the flow set contains the actual data. This arrangement allows for flexible export.

The use of templates with the NetFlow version 9 export format provides several other key benefits:

- You can export almost any information from a router or switch, including Layer 2 through 7 information, routing information, IP version 6 (IPv6), IP version 4 (IPv4), multicast, and Multiprotocol Label Switching (MPLS) information. This new information allows new applications for export data and new views of network behavior.
- Third-party business partners who produce applications that provide NetFlow collector or display services for NetFlow are not required to recompile their applications each time a new NetFlow export field is added. Instead, they can use an external data file that documents the known template formats.
- New features can be added to NetFlow more quickly, without breaking current implementations.
- NetFlow is "future-proofed" against new or developing protocols, because the version 9 export format can be adapted to provide support for them and for other non-NetFlow-based approaches to data collection.

The NetFlow version 9 export packet header format is shown in the figure below.

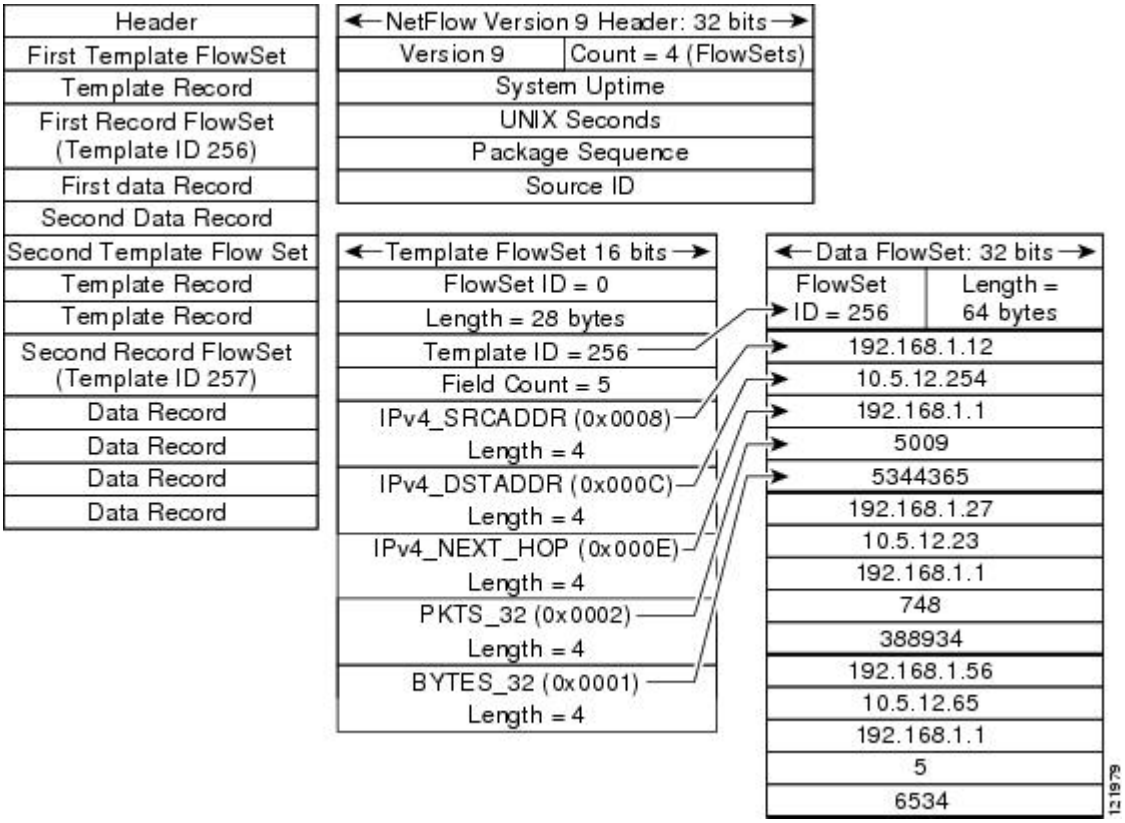
Table 24 NetFlow Version 9 Export Packet Header Field Names and Descriptions

Bytes	Field Name	Description
0-1	Version	The version of NetFlow records exported in this packet; for version 9, this value is 0x0009.

Bytes	Field Name	Description
2-3	Count	Number of FlowSet records (both template and data) contained within this packet.
4-7	System Uptime	Time in milliseconds since this device was first booted.
8-11	UNIX Seconds	Seconds since 0000 Coordinated Universal Time (UTC) 1970.
12-15	Sequence Number	Incremental sequence counter of all export packets sent by this export device; this value is cumulative, and it can be used to find out whether any export packets have been missed. This is a change from the NetFlow version 5 and version 8 headers, where this number represented "total flows."
16-19	Source ID	The Source ID field is a 32-bit value that is used to guarantee uniqueness for each flow exported from a particular device. (The Source ID field is the equivalent of the engine type and engine ID fields found in the NetFlow version 5 and version 8 headers.) The format of this field is vendor specific. In Cisco's implementation, the first two bytes are reserved for future expansion and are always zero. Byte 3 provides uniqueness with respect to the routing engine on the exporting device. Byte 4 provides uniqueness with respect to the particular line card or Versatile Interface Processor on the exporting device. Collector devices should use the combination of the source IP address and the Source ID field to associate an incoming NetFlow export packet with a unique instance of NetFlow on a particular device.

The table below shows a typical example of exporting data using the NetFlow version 9 export format.

Figure 41 ***NetFlow Version 9 Export Format Packet Example***



Additional information about the NetFlow export format version 9 and the export format architecture is available in the NetFlow version 9 Flow-Record Format document.

Configuring Netflow v9 for IPv6

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **mls flow** {**ip** | **ipv6**} {**destination** | **destination-source** | **full** | **interface-destination-source** | **interface-full** | **source**}
5. **mls nde sender**
6. **ip flow-export version 9**
7. **ip flow-export destination** {*ip-address* | *hostname*} *udp-port*
8. **interface** *type number*
9. **ipv6 address** *ip-address/mask*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Router(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.
Step 4	**mls flow** {**ip** | **ipv6**} {**destination** | **destination-source** | **full** | **interface-destination-source** | **interface-full** | **source**} **Example:** `Router(config)# mls flow ipv6 interface-full`	Specifies the NetFlow flow mask for IPv6 traffic.
Step 5	**mls nde sender** **Example:** `Route(config)# mls nde sender`	Enables NDE globally on the router. **Note** NDE does not start exporting data until you specify a destination for the exported traffic. The destination for exported traffic is specified in Step 7.
Step 6	**ip flow-export version 9** **Example:** `Router(config)# ip flow-export version 9`	Configures NDE to use the NetFlow version 9 export format.
Step 7	**ip flow-export destination** {*ip-address* | *hostname*} *udp-port* **Example:** `Router(config)# ip flow-export destination 172.16.10.2 88`	Specifies the IP address or the hostname of the NetFlow collector and the UDP port on which the NetFlow collector is listening.

	Command or Action	Purpose
Step 8	**interface** *type number* **Example:** `Router(config)# interface fastethernet 1/1`	Specifies the interface that you want to enable NetFlow on and enters interface configuration mode.
Step 9	**ipv6 address** *ip-address/mask* **Example:** `Router(config-if)# ipv6 address 2001:0DB8:AB::2/64`	Configure an IPv6 address on the interface.

Examples

The following output of the **show mls nde** command verifies that NDE is enabled on the router.

```
Router# show mls nde

 NetFlow Data Export enabled

 Exporting flows to 10.30.30.2 (12345)  172.16.10.2 (88)

 Exporting flows from 10.4.9.149 (58970)

 Version: 9

 Layer2 flow creation is disabled

 Layer2 flow export is disabled

 Include Filter not configured

 Exclude Filter not configured

 Total NetFlow Data Export Packets are:

    0 packets, 0 no packets, 0 records

 Total NetFlow Data Export Send Errors:

        IPWRITE_NO_FIB = 0

        IPWRITE_ADJ_FAILED = 0

        IPWRITE_PROCESS = 0

        IPWRITE_ENQUEUE_FAILED = 0

        IPWRITE_IPC_FAILED = 0

        IPWRITE_OUTPUT_FAILED = 0

        IPWRITE_MTU_FAILED = 0

        IPWRITE_ENCAPFIX_FAILED = 0

 NetFlow Aggregation Disabled
```

Configuration Examples for Configuring Netflow v9 for IPv6

Example: Configuring the NetFlow v9 for IPv6 Feature

```
ipv6 unicast-routing
```

```
mls flow ipv6 interface-full
mls nde sender
ip flow-export version 9
ip flow-export destination 172.16.10.2 88
interface FastEthernet1/1
ipv6 address
2001:0DB8::1/64
```

Additional References

Related Documents

Related Topic	Document Title
Platform-independent NetFlow commands, complete command syntax, command mode, defaults, command history, usage guidelines, and examples.	*Cisco IOS NetFlow Command Reference*
Command reference for Cisco 7600 series routers	Cisco 7600 Series Cisco IOS Command Reference

Standards

Standard	Title
There are no standards associated with this feature.	--

MIBs

MIB	MIBs Link
•	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 3954	*Cisco Systems NetFlow Services Export Version 9*

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/techsupport

Feature Information for Netflow v9 for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 25 Feature Information for Netflow v9 for IPv6

Feature Name	Releases	Feature Information
Netflow v9 for IPv6	12.2(33)SRB15.0(1)S	The Netflow v9 for IPv6 feature enables the export of NetFlow flow information for IPv6 traffic. In 12.2(33)SRB, support for this feature was introduced on the Cisco 7600 series routers.

Implementing NTPv4 in IPv6

The Network Time Protocol (NTP) is a protocol designed to time-synchronize a network of machines. NTP runs over UDP, which in turn runs over IPv4. NTP Version 4 (NTPv4) is an extension of NTP version 3, which supports both IPv4 and IPv6.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Implementing NTPv4 in IPv6

NTP Version 4

The Network Time Protocol (NTP) is a protocol designed to time-synchronize a network of machines. NTP runs over UDP, which in turn runs over IPv4. NTP Version 4 (NTPv4) is an extension of NTP version 3. NTPv4 supports both IPv4 and IPv6 and is backward-compatible with NTPv3.

NTPv4 provides the following capabilities:

- NTPv4 supports IPv6, making NTP time synchronization possible over IPv6.

- Security is improved over NTPv3. The NTPv4 protocol provides a whole security framework based on public key cryptography and standard X509 certificates.
- Using specific multicast groups, NTPv4 can automatically calculate its time-distribution hierarchy through an entire network. NTPv4 automatically configures the hierarchy of the servers in order to achieve the best time accuracy for the lowest bandwidth cost. This feature leverages site-local IPv6 multicast addresses.

NTPv4 Overview

NTPv4 works in much the same way as does NTP. An NTP network usually gets its time from an authoritative time source, such as a radio clock or an atomic clock attached to a time server. NTP then distributes this time across the network. NTP is extremely efficient; no more than one packet per minute is necessary to synchronize two machines to the accuracy of within a millisecond of one another.

NTP uses the concept of a "stratum" to describe how many NTP "hops" away a machine is from an authoritative time source. A "stratum 1" time server typically has an authoritative time source (such as a radio or atomic clock, or a GPS time source) directly attached, a "stratum 2" time server receives its time via NTP from a "stratum 1" time server, and so on.

NTP avoids synchronizing to a machine whose time may not be accurate in two ways. First, NTP never synchronizes to a machine that is not in turn synchronized itself. Second, NTP compares the time reported by several machines, and will not synchronize to a machine whose time is significantly different than the others, even if its stratum is lower. This strategy effectively builds a self-organizing tree of NTP servers.

The Cisco implementation of NTP does not support stratum 1 service; in other words, it is not possible to connect to a radio or atomic clock (for some specific platforms, however, you can connect a GPS time-source device).

If the network is isolated from the internet, the Cisco implementation of NTP allows a machine to be configured so that it acts as though it is synchronized via NTP, when in fact it has determined the time using other means. Other machines can then synchronize to that machine via NTP.

A number of manufacturers include NTP software for their host systems, and a publicly available version for systems running UNIX and its various derivatives is also available. This software also allows UNIX-derivative servers to acquire the time directly from an atomic clock which would subsequently propagate time information along to Cisco routers.

The communications between machines running NTP (known as "associations") are usually statically configured; each machine is given the IPv4 or IPv6 address of all machines with which it should form associations. Accurate timekeeping is made possible by exchanging NTP messages between each pair of machines with an association.

NTPv4 Features

IPv6 Multicast Mode

NTPv3 supports sending and receiving clock updates using IPv4 broadcast messages. Many network administrators use this feature to distribute time on LANs with minimum client configuration. For example, Cisco corporate LANs use this feature over IPv4 on local gateways. End-user workstations are configured to listen to NTP broadcast messages and synchronize their clocks accordingly.

In NTPv4 for IPv6, IPv6 multicast messages instead of IPv4 broadcast messages are used to send and receive clock updates.

NTP Access Groups versus Symmetric Key Authentication

NTPv3 access group functionality is based on IPv4 numbered access lists. NTPv4 access group functionality accepts IPv6 named access lists as well as IPv4 numbered access lists.

NTP access groups are very useful for assigning NTP permission groups to Cisco IOS access lists. For example, all hosts in a subnet can be allowed to synchronize their clocks from a router but not to provide clock updates to the router. NTP access groups are built on the Cisco IOS access-list infrastructure and deliver fully flexible access-list-based matching functionality.

Although more flexible than NTP symmetric key authentication and easier to deploy, access groups do not provide the same level of security. NTP symmetric key authentication provides a cryptographically strong authentication mechanism, but requires the manual distribution of keys on the NTP devices across the network.

NTP symmetric key authentication is also less flexible than access groups regarding the type of permission that can be associated with different peers. NTP symmetric key authentication is mainly intended for protecting the local router from being updated with wrong clock information from an intruder.

DNS Support for IPv6 in NTPv4

NTPv4 adds DNS support for IPv6. NTPv3 resolves hostnames into IPv4 addresses at configuration (when the command is parsed). Then, only the resolved IPv4 address is kept in memory and stored in NVRAM during NVGEN. The hostname given by the user is lost.

NTPv4 keeps the hostname in memory, so that it can be saved during NVGEN. Configurations saved with hostnames are still readable by NTPv3.

How to Implement NTPv4 in IPv6

NTP services are disabled on all interfaces by default. The following sections contain optional tasks that you can perform on your networking device:

Configuring Poll-Based NTPv4 Associations

Networking devices running NTPv4 can be configured to operate in variety of association modes when synchronizing time with reference time sources. There are two ways that a networking device can obtain time information on a network: by polling host servers and by listening to NTPv4 broadcasts.

The following are two most commonly used poll-based association modes:

- Client mode
- Symmetric active mode

When a networking device is operating in the client mode, it polls its assigned time serving hosts for the current time. The networking device will then pick a host from all the polled time servers to synchronize with. Because the relationship that is established in this case is a client-host relationship, the host will not capture or use any time information sent by the local client device. This mode is most suited for file-server and workstation clients that are not required to provide any form of time synchronization to other local clients. Use the **ntp server** command to individually specify the time serving hosts that you want your networking device to consider synchronizing with and to set your networking device to operate in the client mode.

When a networking device is operating in the symmetric active mode, it polls its assigned time serving hosts for the current time and it responds to polls by its hosts. Because this is a peer-to-peer relationship, the host will also retain time-related information about the local networking device that it is communicating with. This mode should be used when there are several mutually redundant servers that are interconnected using diverse network paths. Most Stratum 1 and stratum 2 servers on the Internet today adopt this form of network setup. Use the **ntp peer** command to specify individually the time serving hosts that you want your networking device to consider synchronizing with and to set your networking device to operate in the symmetric active mode.

The specific mode that you should set each of your networking devices to depends primarily on the role that you want it to assume as a timekeeping device (server or client) and its proximity to a stratum 1 timekeeping server.

- Configuring Symmetric Active Mode, page 590
- Configuring Client Mode, page 591

Configuring Symmetric Active Mode

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ntp peer** {**vrf** *vrf-name* | *ip-address* | *ipv6 address* | **ipv4** | **ipv6** | *hostname*}[**normal-sync**][**version** *number*] [**key** *key-id*] [**source** *interface*] [**prefer**] [**maxpoll** *number*] [**minpoll** *number*] [**burst**] [**iburst**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ntp peer** {**vrf** *vrf-name* | *ip-address* | *ipv6 address* | **ipv4** | **ipv6** | *hostname*} [**normal-sync**][**version** *number*] [**key** *key-id*] [**source** *interface*] [**prefer**] [**maxpoll** *number*] [**minpoll** *number*] [**burst**] [**iburst** **Example:** `Router(config)# ntp peer 2001:DB8:0:0:8:800:200C:417A version 4`	Configures the software clock to synchronize a peer or to be synchronized by a peer.

Configuring Client Mode

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ntp server** {**vrf** *vrf-name* \| *ip-address* \| *ipv6-address* \| **ipv4** \| **ipv6** \| *hostname*}[**normal-sync**][**version** *number*] [**key** *key-id*] [**source** *interface*] [**prefer**] [**maxpoll** *number*] [**minpoll** *number*] [**burst**] [**iburst**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ntp server** {**vrf** *vrf-name* | *ip-address* | *ipv6-address* | **ipv4** | **ipv6** | *hostname*} [**normal-sync**][**version** *number*] [**key** *key-id*] [**source** *interface*] [**prefer**] [**maxpoll** *number*] [**minpoll** *number*] [**burst**] [**iburst** **Example:** `Router(config)# ntp server 2001:DB8:0:0:8:800:200C:417A version 4`	Allows the software clock to be synchronized by an NTP time server.

Configuring Multicast-Based NTPv4 Associations

Configuring an Interface to Send NTPv4 Multicast Packets

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ntp multicast** {*ip-address* | *ipv6-address*} [**key** *key-id*] [**ttl** *value*] [**version** *number*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface fastethernet 0/0`	Specifies an interface type and number, and places the router in interface configuration mode.

	Command or Action	Purpose
Step 4	**ntp multicast** {*ip-address* | *ipv6-address*} [**key** *key-id*] [**ttl** *value*] [**version** *number*] **Example:** `Router(config-if)# ntp multicast FF02::1:FF0E:8C6C`	Configures a system to send NTPv4 multicast packets on a specified interface.

Configuring an Interface to Receive NTPv4 Multicast Packets

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ntp multicast client** {*ip-address* | *ipv6-address*} [**novolley**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface FastEthernet 0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**ntp multicast client** {*ip-address* | *ipv6-address*} [**novolley** **Example:** `Router(config-if)# ntp multicast client FF02::2:FF0E:8C6C`	Configures the system to receive NTP multicast packets on a specified interface.

Defining an NTPv4 Access Group

The access list-based restriction scheme allows you to grant or deny certain access privileges to an entire network, a subnet within a network, or a host within a subnet.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ntp access-group** {**query-only** | **serve-only** | **serve** | **peer**} {*access-list-number* | *access-list-name*} [**kod**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ntp access-group** {**query-only**	**serve-only**	**serve**	**peer**} {*access-list-number*	*access-list-name*} [**kod** **Example:** `Router(config)# ntp access-group serve acl1 kod`	Controls access to the NTPv4 services on the system.

Configuring NTPv4 Authentication

The encrypted NTPv4 authentication scheme should be used when a reliable form of access control is required. Unlike the access list-based restriction scheme, the encrypted authentication scheme uses authentication keys and an authentication process to determine if NTPv4 synchronization packets sent by designated peers or servers on a local network are deemed as trusted before the time information that it carries along with it, is accepted.

After NTPv4 authentication is properly configured, your networking device will only synchronize with and provide synchronization to trusted time sources.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ntp authenticate**
4. **ntp authentication-key** *number* **md5** *value*
5. **ntp trusted-key** *key-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ntp authenticate** **Example:** `Router(config)# ntp authenticate`	Enables NTPv4 authentication.
Step 4	**ntp authentication-key** *number* **md5** *value* **Example:** `Router(config)# ntp authentication-key 42 md5 keyname`	Defines an authentication key for NTPv4.
Step 5	**ntp trusted-key** *key-number* **Example:** `Router(config)# ntp trusted-key 42`	Authenticates the identity of a system to which NTPv4 will synchronize.

Disabling NTPv4 Services on a Specific Interface

NTP and NTPv4 services are disabled on all interfaces by default. NTP or NTPv4 is enabled globally when any NTP commands are entered.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ntp disabl e** [**ipv4** | **ipv6**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ntp disabl e** [**ipv4**	**ipv6** **Example:** `Router(config)# ntp disable ipv6`	Controls access to the NTPv4 services on the system.

Configuring the Source IPv6 Address for NTPv4 Packets

When the system sends an NTPv4 packet, the source IPv6 address is normally set to the address of the interface through which the NTPv4 packet is sent.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ntp source** *type number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ntp source** *type number* **Example:** `Router(config)# ntp source FastEthernet 0/0`	Configures the use of a particular source address in NTPv4 packets. The specified interface is configured with IPv6 addresses.

Configuring the System as an Authoritative NTP Server

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ntp master** [*stratum*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ntp master** [*stratum*] **Example:** `Router(config)# ntp master`	Configures the Cisco IOS software as an NTPv4 master clock to which peers synchronize themselves when an external NTPv4 source is not available.

Note Use the **ntp master**command with caution. It is very easy to override valid time sources using this command, especially if a low stratum number is configured. Configuring multiple machines in the same network with the **ntp master** command can cause instability in timekeeping if the machines do not agree on the time.

Updating the Hardware Clock

On devices that have hardware clocks (system calendars), you can configure the hardware clock to be periodically updated from the software clock. This is advisable for any device using NTPv4, because the time and date on the software clock (set using NTPv4) will be more accurate than the hardware clock, because the time setting on the hardware clock has the potential to drift slightly over time.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ntp update-calendar**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ntp update-calendar** **Example:** `Router(config)# ntp update-calendar`	Periodically updates the hardware clock (calendar) from an NTPv4 time source.

Resetting the Drift Value in the Persistent Data File

The drift is the frequency offset between the local clock hardware and the authoritative time from the Network Time Protocol version 4 (NTPv4) servers. NTPv4 automatically computes this drift and uses it to compensate permanently for local clock imperfections.

SUMMARY STEPS

1. **enable**
2. **ntp drift clear**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**ntp drift clear** **Example:** `Router# ntp drift clear`	Resets the drift value stored in the persistent data file.

Troubleshooting NTPv4 in IPv6

SUMMARY STEPS

1. **enable**
2. **show clock** [**detail**]
3. **show ntp associations** [**detail**
4. **s how ntp status**
5. **debug ntp** {**adjust** | **authentication** | **events** | **loopfilter** | **packets** | **params** | **refclock** | **select** | **sync** | **validity**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show clock** [**detail**] **Example:** `Router# show clock`	Displays the time and date from the system software clock.

	Command or Action	Purpose
Step 3	**show ntp associations** [**detail** **Example:** `Router# show ntp associations`	Shows the status of NTP associations.
Step 4	**s how ntp status** **Example:** `Router# show ntp status`	Shows the status of the NTPv4.
Step 5	**debug ntp** {**adjust** | **authentication** | **events** | **loopfilter** | **packets** | **params** | **refclock** | **select** | **sync** | **validity**} **Example:** `Router# debug ntp`	Displays debugging messages for NTPv4 features.

Configuration Examples for NTPv4 in IPv6

Example: Defining an NTPv4 Access Group

In the following IPv6 example, an NTPv4 access group is enabled and a KOD packet is sent to any host that tries to send a packet that is not compliant with the access-group policy:

```
Router> enable
Router# configure terminal
Router(config)# ntp access-group serve acl1 kod
```

Additional References

Related Documents

Related Topic	Document Title
NTP for IPv4	Performing Basic System Management

Related Topic	Document Title
IPv6 supported feature list	"Start Here: Cisco IOS Software Release Specifics for IPv6 Features ," *Cisco IOS IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*

Standards

Standard	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 1305	*Network Time Protocol (Version 3) Specification, Implementation and Analysis*
RFC 5905	*Network Time Protocol Version 4: Protocol and Algorithms Specification*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing NTPv4 in IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 26 Feature Information for Implementing Selective Packet Discard in IPv6

Feature Name	Releases	Feature Information
IPv6 NTPv4	12.4(20)T 12.2(33)SXJ	The following commands were introduced or modified: **debug ntp**, **ntp access-group**, **ntp authenticate**, **ntp authentication-key**, **ntp broadcast**, **ntp broadcast client**, **ntp broadcastdelay**, **ntp disable**, **ntp drift clear**, **ntp logging**, **ntp master**, **ntp max-associations**, **ntp multicast**, **ntp multicast client**, **ntp peer**, **ntp refclock**, **ntp server**, **ntp source**, **ntp trusted-key**, **ntp update-calendar**, **show clock**, **show ntp associations**, **show ntp status**.

Implementing OSPFv3

This module describes how to implement Open Shortest Path First version 3 (OSPFv3) to provide support for IPv6 routing prefixes.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Implementing OSPFv3

- Complete the OSPFv3 network strategy and planning for your IPv6 network. For example, you must decide whether multiple areas are required.
- Enable IPv6 unicast routing.
- Enable IPv6 on the interface.
- Configure the IP security (IPsec) secure socket application program interface (API) on OSPFv3 in order to enable authentication and encryption.
- Before you can use the IPv4 unicast address families (AFs) in OSPFv3, you must enable IPv6 on a link, although the link may not be participating in IPv6 unicast AF.
- With the OSPFv3 Address Families feature, you may have two device processes per interface, but only one process per AF. If the AF is IPv4, you must first configure an IPv4 address on the interface, but IPv6 must be enabled on the interface.

Restrictions for Implementing OSPFv3

- OSPFv3 can be implemented using either the **ipv6 router ospf** command or the **router ospfv3** command. If you start your configuration using **ipv6 router ospf** commands, you can switch to router ospfv3 configuration mode. However, once you enter router ospfv3 configuration mode you cannot switch back to ipv6 router ospf configuration mode.
- When running a dual-stack IP network with OSPF version 2 for IPv4 and OSPFv3, be careful when changing the defaults for commands used to enable OSPFv3. Changing these defaults may negatively affect your OSPFv3 network.
- Authentication is supported as of Cisco IOS Release 12.3(4)T.
- Encapsulating security payload (ESP) authentication and encryption are supported as of Cisco IOS Release 12.4(9)T.
- A packet will be rejected on a device if the packet is coming from an IPv6 address that is found on any interface on the same device.

Information About Implementing OSPFv3

How OSPFv3 Works

OSPFv3 is a routing protocol for IPv4 and IPv6. It is a link-state protocol, as opposed to a distance-vector protocol. Think of a link as being an interface on a networking device. A link-state protocol makes its routing decisions based on the states of the links that connect source and destination machines. The state of a link is a description of that interface and its relationship to its neighboring networking devices. The interface information includes the IPv6 prefix of the interface, the network mask, the type of network it is connected to, the devices connected to that network, and so on. This information is propagated in various type of link-state advertisements (LSAs).

A device's collection of LSA data is stored in a link-state database. The contents of the database, when subjected to the Dijkstra algorithm, result in the creation of the 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 via specific device interface ports.

OSPFv3, which is described in RFC 5340, supports IPv6 and IPv4 unicast AFs.

Comparison of OSPFv3 and OSPF Version 2

Much of OSPF version 3 is the same as in OSPF version 2. OSPFv3, which is described in RFC 5340, expands on OSPF version 2 to provide support for IPv6 routing prefixes and the larger size of IPv6 addresses.

In OSPFv3, a routing process does not need to be explicitly created. Enabling OSPFv3 on an interface will cause a routing process, and its associated configuration, to be created.

In OSPFv3, each interface must be enabled using commands in interface configuration mode. This feature is different from OSPF version 2, in which interfaces are indirectly enabled using the device configuration mode.

When using a nonbroadcast multiaccess (NBMA) interface in OSPFv3, you must manually configure the device with the list of neighbors. Neighboring devices are identified by their device ID.

In IPv6, you can configure many address prefixes on an interface. In OSPFv3, all address prefixes on an interface are included by default. You cannot select some address prefixes to be imported into OSPFv3; either all address prefixes on an interface are imported, or no address prefixes on an interface are imported.

Unlike OSPF version 2, multiple instances of OSPFv3 can be run on a link.

OSPF automatically prefers a loopback interface over any other kind, and it chooses the highest IP address among all loopback interfaces. If no loopback interfaces are present, the highest IP address in the device is chosen. You cannot tell OSPF to use any particular interface.

OSPFv3 Address Families

The OSPFv3 address families feature enables both IPv4 and IPv6 unicast traffic to be supported. With this feature, you may have two device processes per interface, but only one process per AF. If the IPv4 AF is used, an IPv4 address must first be configured on the interface, but IPv6 must be enabled on the interface. A single IPv4 or IPv6 OSPFv3 process running multiple instances on the same interface is not supported.

If you have an IPv6 network that uses OSPFv3 as its Interior Gateway Protocol (IGP) you may want to use the same IGP to help carry and install IPv4 routes. All devices on this network have an IPv6 forwarding stack. Some (or all) of the links on this network may be allowed to do IPv4 forwarding and be configured with IPv4 addresses. Pockets of IPv4-only devices exist around the edges running an IPv4 static or dynamic routing protocol. In this scenario, you need the ability to forward IPv4 traffic between these pockets without tunneling overhead, which means that any IPv4 transit device has both IPv4 and IPv6 forwarding stacks (that is, dual stack).

This feature allows a separate (possibly incongruent) topology to be constructed for the IPv4 AF. It installs IPv4 routes in the IPv4 Routing Information Base (RIB), and then the forwarding occurs natively. The OSPFv3 process fully supports an IPv4 AF topology and can redistribute routes from and into any other IPv4 routing protocol.

An OSPFv3 process can be configured to be either IPv4 or IPv6. The **address-family** command is used to determine which AF will run in the OSPFv3 process, and only one address family can be configured per instance. Once the AF is selected, you can enable multiple instances on a link and enable address-family-specific commands.

Different instance ID ranges are used for each AF. Each AF establishes different adjacencies, has a different link state database, and computes a different shortest path tree. The AF then installs the routes in the AF-specific RIB. LSAs that carry IPv6 unicast prefixes are used without any modification in different instances to carry each AF's prefixes.

The IPv4 subnets configured on OSPFv3-enabled interfaces are advertised through intra-area prefix LSAs, just as any IPv6 prefixes. External LSAs are used to advertise IPv4 routes redistributed from any IPv4 routing protocol, including connected and static. The IPv4 OSPFv3 process runs the Shortest Path First (SPF) calculations and finds the shortest path to those IPv4 destinations. These computed routes are then inserted in the IPv4 RIB (computed routes are inserted into an IPv6 RIB for an IPv6 AF).

Because the IPv4 OSPFv3 process allocates a unique pdbindex in the IPv4 RIB, all other IPv4 routing protocols can redistribute routes from it. The parse chain for all protocols is the same, so the **ospfv3** keyword added to the list of IPv4 routing protocols causes OSPFv3 to appear in the **redistribute** command from any IPv4 routing protocol. With the **ospfv3** keyword, IPv4 OSPFv3 routes can be redistributed into any other IPv4 routing protocol as defined in the **redistribute ospfv3** command.

The OSPFv3 address families feature is supported as of Cisco IOS Release 15.1(3)S and Cisco IOS Release 15.2(1)T. Cisco devices that run software older than these releases and third-party devices will not neighbor with devices running the AF feature for the IPv4 AF because they do not set the AF bit. Therefore, those devices will not participate in the IPv4 AF SPF calculations and will not install the IPv4 OSPFv3 routes in the IPv6 RIB.

LSA Types for OSPFv3

The following list describes LSA types, each of which has a different purpose:

- Router LSAs (Type 1)—Describes the link state and costs of a router's links to the area. These LSAs are flooded within an area only. The LSA indicates if the router is an Area Border Router (ABR) or Autonomous System Boundary Router (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 when running the SPF calculation.
- Network LSAs (Type 2)—Describes the link-state and cost information for all routers attached to the network. This LSA is an aggregation of all the link-state and cost information in the network. Only a designated router tracks this information and can generate a network LSA. In OSPFv3, network LSAs have no address information and are network-protocol-independent.
- Interarea-prefix LSAs 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 summarized into one advertisement. Only ABRs generate summary LSAs. In OSPFv3, addresses for these LSAs are expressed as *prefix*, *prefix length* instead of *address*, *mask*. The default route is expressed as a prefix with length 0.
- Interarea-router LSAs for ASBRs (Type 4)—Advertises the location of an ASBR. Routers that are trying to reach an external network use these advertisements to determine the best path to the next hop. Type 4 LSAs are generated by ABRs on behalf of ASBRs.
- Autonomous system external LSAs (Type 5)—Redistributes routes from another autonomous system, usually from a different routing protocol into OSPFv3. In OSPFv3, addresses for these LSAs are expressed as *prefix*, *prefix length* instead of *address*, *mask*. The default route is expressed as a prefix with length 0.
- Link LSAs (Type 8)—Have local-link flooding scope and are never flooded beyond the link with which they are associated. Link LSAs provide the link-local address of the router to all other routers attached to the link, inform other routers attached to the link of a list of prefixes to associate with the link, and allow the router to assert a collection of Options bits to associate with the network LSA that will be originated for the link.
- Intra-Area-Prefix LSAs (Type 9)—A router can originate multiple intra-area-prefix LSAs for each 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 the 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: PrefixLength, PrefixOptions, and Address Prefix. In OSPFv3, addresses for these LSAs are expressed as *prefix, prefix length* instead of *address, mask*. The default route is expressed as a prefix with length 0. Type 3 and Type 9 LSAs carry all prefix (subnet) information that, in OSPFv2, 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 OSPFv3.

In OSPFv3, the sole function of the 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.

The link-state ID in network LSAs and link LSAs is always the interface ID of the originating router on the link being described. For this reason, network LSAs and link LSAs are now the only LSAs whose size cannot be limited. A network LSA must list all routers connected to the link, and a link LSA must list all of the address prefixes of a router on the link.

- OSPFv3 Max-Metric Router LSA, page 607

OSPFv3 Max-Metric Router LSA

The OSPFv3 max-metric router LSA feature enables OSPFv3 to advertise its locally generated router LSAs with a maximum metric. The feature allows OSPFv3 processes to converge but not attract transit traffic through the device if there are better alternate paths. After a specified timeout or a notification from Border Gateway Protocol (BGP), OSPFv3 advertises the LSAs with normal metrics.

The max-metric LSA control places the OSPFv3 router into the stub router role using its LSA advertisement. A stub router only forwards packets destined to go to its directly connected links. In OSPFv3 networks, a device could become a stub router by advertising large metrics for its connected links, so that the cost of a path through this device becomes larger than that of an alternative path. OSPFv3 stub router advertisement allows a device to advertise the infinity metric (0xFFFF) for its connected links in router LSAs and advertise the normal interface cost if the link is a stub network.

NBMA in OSPFv3

On NBMA networks, the designated router (DR) or backup DR (BDR) performs the LSA flooding. On point-to-point networks, flooding simply goes out an interface directly to a neighbor.

Devices that share a common segment (Layer 2 link between two interfaces) become neighbors on that segment. OSPFv3 uses the Hello protocol, periodically sending hello packets out each interface. Devices become neighbors when they see themselves listed in the neighbor’s hello packet. After two devices become neighbors, they may proceed to exchange and synchronize their databases, which creates an adjacency. Not all neighboring devices have an adjacency.

On point-to-point and point-to-multipoint networks, the software floods routing updates to immediate neighbors. There is no DR or BDR; all routing information is flooded to each networking device.

On broadcast or NBMA segments only, OSPFv3 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 devices on the segment have a central point of contact for information exchange. Instead of each device exchanging routing updates with every other device on the segment, each device exchanges information with the DR and BDR. The DR and BDR relay the information to the other devices.

The software looks at the priority of the devices on the segment to determine which devices will be 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 device with a router priority set to zero is ineligible to become the DR or BDR.

When using NBMA in OSPFv3, you cannot automatically detect neighbors. On an NBMA interface, you must configure your neighbors manually using interface configuration mode.

Force SPF in OSPFv3

When the **process** keyword is used with the **clear ipv6 ospf** command, the OSPFv3 database is cleared and repopulated, and then the SPF algorithm is performed. When the **force-spf** keyword is used with the **clear ipv6 ospf** command, the OSPFv3 database is not cleared before the SPF algorithm is performed.

Fast Convergence: LSA and SPF Throttling

The OSPFv3 LSA and SPF throttling feature provides a dynamic mechanism to slow down link-state advertisement updates in OSPFv3 during times of network instability. It also allows faster OSPFv3 convergence by providing LSA rate limiting in milliseconds.

OSPFv3 can use static timers for rate-limiting SPF calculation and LSA generation. Although these timers are configurable, the values used are specified in seconds, which poses a limitation on OSPFv3 convergence. LSA and SPF throttling achieves subsecond convergence by providing a more sophisticated SPF and LSA rate-limiting mechanism that is able to react quickly to changes and also provide stability and protection during prolonged periods of instability.

Load Balancing in OSPFv3

When a device learns multiple routes to a specific network via multiple routing processes (or routing protocols), it installs the route with the lowest administrative distance in the routing table. Sometimes the device must select a route from among many learned via the same routing process with the same administrative distance. In this case, the device chooses the path with the lowest cost (or metric) to the destination. Each routing process calculates its cost differently and the costs may need to be manipulated in order to achieve load balancing.

OSPFv3 performs load balancing automatically in the following way. If OSPFv3 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** command. The default maximum paths is 16, and the range is from 1 to 64.

Addresses Imported into OSPFv3

When importing the set of addresses specified on an interface on which OSPFv3 is running into OSPFv3, you cannot select specific addresses to be imported. Either all addresses are imported, or no addresses are imported.

OSPFv3 Customization

You can customize OSPFv3 for your network, but you likely will not need to do so. The defaults for OSPFv3 are set to meet the requirements of most customers and features. If you must change the defaults, refer to the IPv6 command reference to find the appropriate syntax.

Caution Be careful when changing the defaults. Changing defaults will affect your OSPFv3 network, possibly adversely.

OSPFv3 Authentication Support with IPsec

In order to ensure that OSPFv3 packets are not altered and re-sent to the router, causing the router to behave in a way not desired by its system administrators, OSPFv3 packets must be authenticated. OSPFv3 uses the IPsec secure socket API to add authentication to OSPFv3 packets. This API supports IPv6.

OSPFv3 requires the use of IPsec to enable authentication. Crypto images are required to use authentication, because only crypto images include the IPsec API needed for use with OSPFv3.

In OSPFv3, authentication fields have been removed from OSPFv3 packet headers. When OSPFv3 runs on IPv6, OSPFv3 requires the IPv6 authentication header (AH) or IPv6 ESP header to ensure integrity, authentication, and confidentiality of routing exchanges. IPv6 AH and ESP extension headers can be used to provide authentication and confidentiality to OSPFv3.

To use the IPsec AH, you must enable the **ipv6 ospf authentication** command. To use the IPsec ESP header, you must enable the **ipv6 ospf encryption** command. The ESP header may be applied alone or in combination with the AH, and when ESP is used, both encryption and authentication are provided. Security services can be provided between a pair of communicating hosts, between a pair of communicating security gateways, or between a security gateway and a host.

To configure IPsec, you configure a security policy, which is a combination of the security policy index (SPI) and the key (the key is used to create and validate the hash value). IPsec for OSPFv3 can be configured on an interface or on an OSPFv3 area. For higher security, you should configure a different policy on each interface configured with IPsec. If you configure IPsec for an OSPFv3 area, the policy is applied to all of the interfaces in that area, except for the interfaces that have IPsec configured directly. Once IPsec is configured for OSPFv3, IPsec is invisible to you.

The secure socket API is used by applications to secure traffic. The API needs to allow the application to open, listen, and close secure sockets. The binding between the application and the secure socket layer also allows the secure socket layer to inform the application of changes to the socket, such as connection open and close events. The secure socket API is able to identify the socket; that is, it can identify the local and remote addresses, masks, ports, and protocol that carry the traffic requiring security.

Each interface has a secure socket state, which can be one of the following:

- NULL: Do not create a secure socket for the interface if authentication is configured for the area.
- DOWN: IPsec has been configured for the interface (or the area that contains the interface), but OSPFv3 either has not requested IPsec to create a secure socket for this interface, or there is an error condition.
- GOING UP: OSPFv3 has requested a secure socket from IPsec and is waiting for a CRYPTO_SS_SOCKET_UP message from IPsec.
- UP: OSPFv3 has received a CRYPTO_SS_SOCKET_UP message from IPsec.
- CLOSING: The secure socket for the interface has been closed. A new socket may be opened for the interface, in which case the current secure socket makes the transition to the DOWN state. Otherwise, the interface will become UNCONFIGURED.
- UNCONFIGURED: Authentication is not configured on the interface.

OSPFv3 will not send or accept packets while in the DOWN state.

- OSPFv3 Virtual Links, page 610

OSPFv3 Virtual Links

For each virtual link, a master security information datablock is created for the virtual link. Because a secure socket must be opened on each interface, there will be a corresponding security information datablock for each interface in the transit area. The secure socket state is kept in the interface's security information datablock. The state field in the master security information datablock shows the status of all of the secure sockets opened for the virtual link. If all of the secure sockets are UP, then the security state for the virtual link will be set to UP.

Packets sent on a virtual link with IPsec must use predetermined source and destination addresses. The first local area address found in the router's intra-area-prefix LSA for the area is used as the source address. This source address is saved in the area data structure and used when secure sockets are opened and packets sent over the virtual link. The virtual link will not transition to the point-to-point state until a source address is selected. Also, when the source or destination address changes, the previous secure sockets must be closed and new secure sockets opened.

Note Virtual links are not supported for the IPv4 AF.

OSPFv3 Cost Calculation

Because cost components can change rapidly, it might be necessary to reduce the volume of changes to reduce network-wide churn. The recommended values for S2, S3, and S4 in the second table below are based on network simulations that may reduce the rate of network changes. The recommended value for S1 is 0 to eliminate this variable from the route cost calculation.

The overall link cost is computed using the formula shown in the figure below.

Figure 42 Overall Link Cost Formula

$$\text{LinkCost} = \text{OC} + \text{BW}\left(\frac{\text{Throughput\_weight}}{100}\right) + \text{Resources}\left(\frac{\text{Resources\_weight}}{100}\right) + \text{Latency}\left(\frac{\text{Latency\_weight}}{100}\right) + \text{L2\_factor}\left(\frac{\text{L2\_weight}}{100}\right)$$

$$\text{OC} = \left[\frac{(\text{ospf\_reference\_bw})}{(\text{MDR})(1000)}\right] \qquad \text{ospf\_reference\_bw} = 10 \wedge 8$$

$$\text{BW} = \frac{(65535)\left(100 - \frac{\text{CDR}}{\text{MDR}}(100)\right)}{100}$$

$$\text{Resources} = \frac{(100 - \text{resources})^3 (65535)}{1000000}$$

$$\text{Latency} = \text{latency}$$

$$\text{L2\_factor} = \frac{(100 - \text{RLQ})(65535)}{100}$$

231048

The table below defines the symbols used in the OSPFv3 cost calculation.

Table 27 OSPFv3 Cost Calculation Definitions

Cost Component	Component Definition
OC	The default OSPFv3 cost. Calculated from reference bandwidth using reference_bw / (MDR*1000), where reference_bw=10^8.
A through D	Various radio-specific data-based formulas that produce results in the 0 through 64,000 range.
A	CDR- and MDR-related formula: (2^16 * (100 – (CDR * 100 / MDR)))/100
B	Resources related formula: ((100 – RESOURCES)^3 * 2^16 / 10^6)
C	Latency as reported by the radio, already in the 0 through 64,000 range when reported (LATENCY).
D	RLF-related formula: ((100 – RLF) * 2^16)/100
S1 through S4	Scalar weighting factors input from the CLI. These scalars scale down the values as computed by A through D. The value of 0 disables and the value of 100 enables full 0 through 64,000 range for one component.

Because each network might have unique characteristics that require different settings to optimize actual network performance, these are recommended values intended as a starting point for optimizing an OSPFv3 network. The table below lists the recommended value settings for OSPFv3 cost metrics.

Table 28 Recommended Value Settings for OSPFv3 Cost Metrics

Setting	Metric Description	Default Value	Recommended Value
S1	ipv6 ospf dynamic weight throughout	100	0
S2	ipv6 ospf dynamic weight resources	100	29
S3	ipv6 ospf dynamic weight latency	100	29
S4	ipv6 ospf dynamic weight L2 factor	100	29

The default path costs were calculated using this formula, as noted in the following list. If these values do not suit your network, you can use your own method of calculating path costs.

- 56-kbps serial link—Default cost is 1785.
- 64-kbps serial link—Default cost is 1562.
- T1 (1.544-Mbps serial link)—Default cost is 64.
- E1 (2.048-Mbps serial link)—Default cost is 48.
- 4-Mbps Token Ring—Default cost is 25.
- Ethernet—Default cost is 10.
- 16-Mbps Token Ring—Default cost is 6.
- FDDI—Default cost is 1.
- X25—Default cost is 5208.
- Asynchronous—Default cost is 10,000.
- ATM—Default cost is 1.

To illustrate these settings, the following example shows how OSPFv3 cost metrics might be defined for a Virtual Multipoint Interface (VMI) interface:

```
interface vmi1
 ipv6 ospf cost dynamic weight throughput 0
 ipv6 ospf cost dynamic weight resources 29
 ipv6 ospf cost dynamic weight latency 29
 ipv6 ospf cost dynamic weight L2-factor 29
```

OSPFv3 External Path Preference Option

Per RFC 5340, the following rules indicate which paths are preferred when multiple intra-AS paths are available to ASBRs or forwarding addresses:

- Intra-area paths using nonbackbone areas are always the most preferred.
- The other paths, intraarea backbone paths and interarea paths, are of equal preference.

These rules apply when the same ASBR is reachable through multiple areas, or when trying to decide which of several AS-external-LSAs should be preferred. In the former case the paths all terminate at the same ASBR, and in the latter the paths terminate at separate ASBRs or forwarding addresses. In either case, each path is represented by a separate routing table entry. This feature applies only when RFC 1583 compatibility is set to disabled using the **no compatibility rfc1583** command (RFC 5340 provides an update to RFC 1583).

Caution To minimize the chance of routing loops, set identical RFC compatibility for all OSPF routers in an OSPF routing domain.

OSPFv3 Graceful Restart

The graceful restart feature in OSPFv3 allows nonstop data forwarding along routes that are already known while the OSPFv3 routing protocol information is being restored. A device can participate in graceful restart either in restart mode (such as in a graceful-restart-capable router) or in helper mode (such as in a graceful-restart-aware router).

To perform the graceful restart function, a device must be in high availability (HA) stateful switchover (SSO) mode (that is, dual Route Processor (RP)). A device capable of graceful restart will perform the graceful restart function when the following failures occur:

- A RP failure that results in switchover to standby RP

- A planned RP switchover to standby RP

The graceful restart feature requires that neighboring devices be graceful-restart aware.

For further information about SSO and nonstop forwarding (NSF), see the Stateful Switchover and Cisco Nonstop Forwarding documents.

BFD Support for OSPFv3

Bidirectional Forwarding Detection (BFD) supports OSPFv3.

How to Implement OSPFv3

Configuring the OSPFv3 Router Process

Once you have completed step 3 and entered OSPFv3 router configuration mode, you can perform any of the subsequent steps in this task as needed to configure OSPFv3 router configuration.

This task can be performed in Cisco IOS Release 15.1(3)S and 15.2(1)T and later releases.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router ospfv3** [*process-id*]
4. **area** *area-ID* [**default-cost** | **nssa** | **stub**]
5. **auto-cost reference-bandwidth** *Mbps*
6. **bfd all-interfaces**
7. **default** {**area** *area-ID* [**range** *ipv6-prefix* | **virtual-link** *router-id*]} [**default-information originate** [**always** | **metric** | **metric-type** | **route-map**] | **distance** | **distribute-list** *prefix-list prefix-list-name* {**in** | **out**} [*interface*] | **maximum-paths** *paths* | **redistribute** *protocol* | **summary-prefix** *ipv6-prefix*]
8. **ignore lsa mospf**
9. **interface-id snmp-if-index**
10. **log-adjacency-changes** [**detail**]
11. **passive-interface** [**default** | *interface-type interface-number*]
12. **queue-depth** {**hello** | **update**} {*queue-size* | **unlimited**}
13. **router-id** *router-id*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router ospfv3** [*process-id*] **Example:** `Device(config)# router ospfv3 1`	Enters router configuration mode for the IPv4 or IPv6 address family.
Step 4	**area** *area-ID* [**default-cost**	**nssa**	**stub**] **Example:** `Device(config-router)# area 1`	Configures the OSPFv3 area.

	Command or Action	Purpose
Step 5	**auto-cost reference-bandwidth** *Mbps* **Example:** `Device(config-router)# auto-cost reference-bandwidth 1000`	Controls the reference value OSPFv3 uses when calculating metrics for interfaces in an IPv4 OSPFv3 process.
Step 6	**bfd all-interfaces** **Example:** `Device(config-router)# bfd all-interfaces`	Enables BFD for an OSPFv3 routing process
Step 7	**default** {**area** *area-ID* [**range** *ipv6-prefix* | **virtual-link** *router-id*]} [**default-information originate** [**always** | **metric** | **metric-type** | **route-map**] | **distance** | **distribute-list** *prefix-list prefix-list-name* {**in** | **out**} [*interface*] | **maximum-paths** *paths* | **redistribute** *protocol* | **summary-prefix** *ipv6-prefix*] **Example:** `Device(config-router)# default area 1`	Returns an OSPFv3 parameter to its default value.
Step 8	**ignore lsa mospf** **Example:** `Device(config-router)# ignore lsa mospf`	Suppresses the sending of syslog messages when the device receives LSA Type 6 multicast OSPFv3 packets, which are unsupported.
Step 9	**interface-id snmp-if-index** **Example:** `Device(config-router)# interface-id snmp-if-index`	Configures OSPFv3 interfaces with Simple Network Management Protocol (SNMP) MIB-II interface Index (ifIndex) identification numbers in IPv4 and IPv6.
Step 10	**log-adjacency-changes** [**detail**] **Example:** `Device(config-router)# log-adjacency-changes`	Configures the device to send a syslog message when an OSPFv3 neighbor goes up or down.
Step 11	**passive-interface** [**default** | *interface-type interface-number*] **Example:** `Device(config-router)# passive-interface default`	Suppresses sending routing updates on an interface when an IPv4 OSPFv3 process is used.

	Command or Action	Purpose
Step 12	**queue-depth** {**hello** | **update**} {*queue-size* | **unlimited**} **Example:** `Device(config-router)# queue-depth update 1500`	Configures the number of incoming packets that the IPv4 OSPFv3 process can keep in its queue.
Step 13	**router-id** *router-id* **Example:** `Device(config-router)# router-id 10.1.1.1`	Enter this command to use a fixed router ID.

Configuring the IPv6 Address Family in OSPFv3

Perform this task to configure the IPv6 address family in OSPFv3. Once you have completed step 4 and entered IPv6 address-family configuration mode, you can perform any of the subsequent steps in this task as needed to configure the IPv6 AF.

This task can be performed in Cisco IOS Release 15.1(3)S and 15.2(1)T and later releases.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router ospfv3** [*process-id*]
4. **address-family ipv6 unicast**
5. **area** *area-ID* **range** *ipv6-prefix* / *prefix-length*
6. **default** {**area** *area-ID* [**range** *ipv6-prefix* | **virtual-link** *router-id*]} [**default-information originate** [**always** | **metric** | **metric-type** | **route-map**] | **distance** | **distribute-list** *prefix-list prefix-list-name* {**in** | **out**} [*interface*] | **maximum-paths** *paths* | **redistribute** *protocol* | **summary-prefix** *ipv6-prefix*]
7. **default-information originate** [**always**] **metric** *metric-value* | **metric-type** *type-value* | **route-map** *map-name*
8. **default-metric** *metric-value*
9. **distance** *distance*
10. **distribute-list prefix-list** *list-name* {**in** [*interface-type interface-number*] | **out** *routing-process* [*as-number*]}
11. **maximum-paths** *number-paths*
12. **summary-prefix** *prefix* [**not-advertise** | **tag** *tag-value*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router ospfv3** [*process-id*] **Example:** `Device(config)# router ospfv3 1`	Enables OSPFv3 router configuration mode for the IPv4 or IPv6 address family.

Step 4	**address-family ipv6 unicast**
	
	Example:
	`Device(config-router)# address-family ipv6 unicast`
	
	Example:
	`Device(config-router)# address-family ipv4 unicast`
		or
		address-family ipv4 unicast
		Enters IPv6 address family configuration mode for OSPFv3.
		or
		Enters IPv4 address family configuration mode for OSPFv3.

Step 5	**area** *area-ID* **range** *ipv6-prefix* / *prefix-length* **Example:** `Device(config-router-af)# area 1 range 2001:DB8:0:0::0/128`	Configures OSPFv3 area parameters.
Step 6	**default** {**area** *area-ID* [**range** *ipv6-prefix* | **virtual-link** *router-id*]} [**default-information originate** [**always** | **metric** | **metric-type** | **route-map**] | **distance** | **distribute-list** *prefix-list prefix-list-name* {**in** | **out**} [*interface*] | **maximum-paths** *paths* | **redistribute** *protocol* | **summary-prefix** *ipv6-prefix*] **Example:** `Device(config-router-af)# default area 1`	Returns an OSPFv3 parameter to its default value.

	Command or Action	Purpose
Step 7	**default-information originate** [**always**] **metric** *metric-value* | **metric-type** *type-value* | **route-map** *map-name* **Example:** `Device(config-router-af)# default-information originate always metric 100 metric-type 2`	Generates a default external route into an OSPFv3 for a routing domain.
Step 8	**default-metric** *metric-value* **Example:** `Device(config-router-af)# default-metric 10`	Sets default metric values for IPv4 and IPv6 routes redistributed into the OSPFv3 routing protocol.
Step 9	**distance** *distance* **Example:** `Device(config-router-af)# distance 200`	Configures an administrative distance for OSPFv3 routes inserted into the routing table.
Step 10	**distribute-list prefix-list** *list-name* {**in** [*interface-type interface-number*] | **out** *routing-process* [*as-number*]} **Example:** `Device(config-router-af)# distribute-list prefix-list PL1 in Ethernet0/0`	Applies a prefix list to OSPFv3 routing updates that are received or sent on an interface.
Step 11	**maximum-paths** *number-paths* **Example:** `Device(config-router-af)# maximum-paths 4`	Controls the maximum number of equal-cost routes that a process for OSPFv3 routing can support.
Step 12	**summary-prefix** *prefix* [**not-advertise** | **tag** *tag-value*] **Example:** `Device(config-router-af)# summary-prefix FEC0::/24`	Configures an IPv6 summary prefix in OSPFv3.

Configuring the IPv4 Address Family in OSPFv3

Perform this task to configure the IPv4 address family in OSPFv3. Once you have completed step 4 and entered IPv4 address family configuration mode, you can perform any of the subsequent steps in this task as needed to configure the IPv4 AF.

This task can be performed in Cisco IOS Release 15.1(3)S and 15.2(1)T and later releases.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router ospfv3** [*process-id*]
4. **address-family ipv4 unicast**
5. **area** *area-id* **range** *ip-address ip-address-mask* [**advertise** | **not-advertise**] [**cost** *cost*]
6. **default** {**area** *area-ID*[**range** *ipv6-prefix* | **virtual-link** *router-id*]} [**default-information originate** [**always** | **metric** | **metric-type** | **route-map**] | **distance** | **distribute-list** *prefix-list prefix-list-name* {**in** | **out**} [*interface*] | **maximum-paths** *paths* | **redistribute** *protocol* | **summary-prefix** *ipv6-prefix*]
7. **default-information originate** [**always**] **metric** *metric-value* | **metric-type** *type-value* | **route-map** *map-name*
8. **default-metric** *metric-value*
9. **distance** *distance*
10. **distribute-list prefix-list** *list-name* {**in** [*interface-type interface-number*] | **out** *routing-process* [*as-number*]}
11. **maximum-paths** *number-paths*
12. **summary-prefix** *prefix* [**not-advertise** | **tag** *tag-value*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router ospfv3** [*process-id*] **Example:** `Device(config)# router ospfv3 1`	Enables OSPFv3 router configuration mode for the IPv4 or IPv6 address family.
Step 4	**address-family ipv4 unicast** **Example:** `Device(config-router)# address-family ipv4 unicast`	Enters IPv4 address family configuration mode for OSPFv3.

	Command or Action	Purpose
Step 5	**area** *area-id* **range** *ip-address ip-address-mask* [**advertise** | **not-advertise**] [**cost** *cost*] **Example:** `Device(config-router-af)# area 0 range 192.168.110.0 255.255.0.0`	Consolidates and summarizes routes at an area boundary.
Step 6	**default** {**area** *area-ID*[**range** *ipv6-prefix* | **virtual-link** *router-id*]} [**default-information originate** [**always** | **metric** | **metric-type** | **route-map**] | **distance** | **distribute-list** *prefix-list prefix-list-name* {**in** | **out**} [*interface*] | **maximum-paths** *paths* | **redistribute** *protocol* | **summary-prefix** *ipv6-prefix*] **Example:** `Device(config-router-af)# default area 1`	Returns an OSPFv3 parameter to its default value.
Step 7	**default-information originate** [**always**] **metric** *metric-value* | **metric-type** *type-value* | **route-map** *map-name* **Example:** `Device(config-router-af)# default-information originate always metric 100 metric-type 2`	Generates a default external route into an OSPFv3 for a routing domain.
Step 8	**default-metric** *metric-value* **Example:** `Device(config-router-af)# default-metric 10`	Sets default metric values for IPv4 and IPv6 routes redistributed into the OSPFv3 routing protocol.
Step 9	**distance** *distance* **Example:** `Device(config-router-af)# distance 200`	Configures an administrative distance for OSPFv3 routes inserted into the routing table.
Step 10	**distribute-list prefix-list** *list-name* {**in** [*interface-type interface-number*] | **out** *routing-process* [*as-number*]} **Example:** `Device(config-router-af)# distribute-list prefix-list PL1 in Ethernet 0/0`	Applies a prefix list to OSPFv3 routing updates that are received or sent on an interface.

	Command or Action	Purpose
Step 11	**maximum-paths** *number-paths* **Example:** `Device(config-router-af)# maximum-paths 4`	Controls the maximum number of equal-cost routes that a process for OSPFv3 routing can support.
Step 12	**summary-prefix** *prefix* [**not-advertise** | **tag** *tag-value*] **Example:** `Device(config-router-af)# summary-prefix FEC0::/24`	Configures an IPv6 summary prefix in OSPFv3.

Configuring Route Redistribution in OSPFv3

This task can be performed in Cisco IOS Release 15.1(3)S and 15.2(1)T and later releases.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router ospfv3** [*process-id*]
4. **address-family ipv6 unicast**
5. **redistribute** source-*protocol* [*process-id*] [*options*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**router ospfv3** [*process-id*] **Example:** `Device(config)# router ospfv3 1`	Enables OSPFv3 router configuration mode for the IPv4 or IPv6 address family.

Step 4	**address-family ipv6 unicast**
	
	Example:
	`Device(config-router)# address-family ipv6 unicast`
	
	Example:
	`Device(config-router)# address-family ipv4 unicast`
		Enters IPv6 address family configuration mode for OSPFv3.
		or
		Enters IPv4 address family configuration mode for OSPFv3.

Step 5	**redistribute** source-*protocol* [*process-id*] [*options*] **Example:**	Redistributes IPv6 and IPv4 routes from one routing domain into another routing domain.

Enabling OSPFv3 on an Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. Do one of the following:
   - **ospfv3** *process-id* **area** *area-ID* {**ipv4** | **ipv6**} [**instance** *instance-id*]
   - **ipv6 ospf** *process-id* **area** *area-id* [**instance** *instance-id*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface ethernet 0/0`	Specifies an interface type and number, and places the device in interface configuration mode.

Step 4	Do one of the following:
	• **ospfv3** *process-id* **area** *area-ID* {**ipv4** | **ipv6**} [**instance** *instance-id*]
	• **ipv6 ospf** *process-id* **area** *area-id* [**instance** *instance-id*]
	
	Example:
	`Device(config-if)# ospfv3 1 area 1 ipv4`
	
	Example:
	`Device(config-if)# ipv6 ospf 1 area 0`
		Enables OSPFv3 on an interface with the IPv4 or IPv6 AF.
		or
		Enables OSPFv3 on an interface.

Defining an OSPFv3 Area Range for the IPv6 or IPv4 Address Family

The cost of the summarized routes will be the highest cost of the routes being summarized. For example, if the following routes are summarized:

```
OI  2001:DB8:0:7::/64 [110/20]
     via FE80::A8BB:CCFF:FE00:6F00, Ethernet0/0
OI  2001:DB8:0:8::/64 [110/100]
     via FE80::A8BB:CCFF:FE00:6F00, Ethernet0/0
OI  2001:DB8:0:9::/64 [110/20]
     via FE80::A8BB:CCFF:FE00:6F00, Ethernet0/0
```

They become one summarized route, as follows:

```
OI  2001:DB8::/48 [110/100]
     via FE80::A8BB:CCFF:FE00:6F00, Ethernet0/0
```

The task can be performed in Cisco IOS Release 15.1(3)S and 15.2(1)T and later releases.

OSPFv3 routing must be enabled.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router ospfv3** [*process-id*]
4. **address-family ipv6 unicast**
5. **area** *area-ID* **range** *ipv6-prefix*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router ospfv3** [*process-id*] **Example:** `Device(config)# router ospfv3 1`	Enables OSPFv3 router configuration mode for the IPv4 or IPv6 address family.

Step 4	**address-family ipv6 unicast**
	
	Example:
	`Device(config-router)# address-family ipv6 unicast`
	
	Example:
	`Device(config-router)# address-family ipv4 unicast`
		Enters IPv6 address family configuration mode for OSPFv3.
		or
		Enters IPv4 address family configuration mode for OSPFv3.

Step 5	**area** *area-ID* **range** *ipv6-prefix* **Example:** `Device(config-router-af)# area 1 range 2001:DB8:0:0::0/128`	Configures OSPFv3 area parameters.

- Defining an OSPFv3 Area Range, page 625

Defining an OSPFv3 Area Range

The task can be performed in releases prior to Cisco IOS Release 15.1(3)S and 15.2(1)T.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router ospf** *process-id*
4. **area** *area-id* **range** *ipv6-prefix* / *prefix-length* [**advertise** | **not-advertise**] [**cost** *cost*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router ospf** *process-id* **Example:** `Device(config)# ipv6 router ospf 1`	Enables OSPFv3 router configuration mode.
Step 4	**area** *area-id* **range** *ipv6-prefix* / *prefix-length* [**advertise** | **not-advertise**] [**cost** *cost*] **Example:** `Device(config-router)# area 1 range 2001:DB8::/48`	Consolidates and summarizes routes at an area boundary.

Configuring the OSPFv3 Max-Metric Router LSA

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router ospfv3** *process-id*
4. **max-metric router-lsa** [**external-lsa** [*max-metric-value*]] [**include-stub**] [**inter-area-lsas** [*max-metric-value*]] [**on-startup** {*seconds* | **wait-for-bgp**}] [**prefix-lsa**] [**stub-prefix-lsa** [*max-metric-value*]] [**summary-lsa** [*max-metric-value*]]
5. **exit**
6. **show ospfv3** [*process-id*] **max-metric**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router ospfv3** *process-id* **Example:** `Device(config)# router ospfv3 1`	Enables OSPFv3 router configuration mode.
Step 4	**max-metric router-lsa** [**external-lsa** [*max-metric-value*]] [**include-stub**] [**inter-area-lsas** [*max-metric-value*]] [**on-startup** {*seconds* | **wait-for-bgp**}] [**prefix-lsa**] [**stub-prefix-lsa** [*max-metric-value*]] [**summary-lsa** [*max-metric-value*]] **Example:** `Device(config-router)# max-metric router-lsa on-startup wait-for-bgp`	Configures a device that is running the OSPFv3 protocol to advertise a maximum metric so that other devices do not prefer the device as an intermediate hop in their SPF calculations.

	Command or Action	Purpose
Step 5	**exit** **Example:** `Device(config-router)# exit`	Leaves the current configuration mode. • Enter this command twice to reach privileged EXEC mode.
Step 6	**show ospfv3** [*process-id*] **max-metric** **Example:** `Device# show ospfv3 1 max-metric`	Displays OSPFv3 maximum metric origination information.

Configuring IPsec on OSPFv3

Once you have configured OSPFv3 and decided on your authentication, you must define the security policy on each of the routers within the group. The security policy consists of the combination of the key and the SPI. To define a security policy, you must define an SPI and a key.

You can configure an authentication or encryption policy either on an interface or for an OSPFv3 area. When you configure for an area, the security policy is applied to all of the interfaces in the area. For higher security, use a different policy on each interface.

You can configure authentication and encryption on virtual links.

Defining Authentication on an Interface

Before you configure IPsec on an interface, you must configure OSPFv3 on that interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. Do one of the following:
   - **ospfv3 authentication** {**ipsec** *spi*} {**md5** | **sha1**}{ *key-encryption-type key*} | **null**
   - **ipv6 ospf authentication** {**null** | **ipsec spi** *spi authentication-algorithm* [*key-encryption-type*] [*key*]}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface ethernet 0/0`	Specifies an interface type and number, and places the device in interface configuration mode.

Step 4	Do one of the following:
	• **ospfv3 authentication** {**ipsec** *spi*} {**md5** | **sha1**}{ *key-encryption-type key*} | **null**
	• **ipv6 ospf authentication** {**null** | **ipsec spi** *spi authentication-algorithm* [*key-encryption-type*] [*key*]}
	
	Example:
	`Device(config-if)# ospfv3 authentication md5 0 27576134094768132473302031209727`
	
	Example:
	Or
	
	`Device(config-if)# ipv6 ospf authentication ipsec spi 500 md5 1234567890abcdef1234567890abcdef`
		Specifies the authentication type for an interface.

Defining Encryption on an Interface

Before you configure IPsec on an interface, you must configure OSPFv3 on that interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. Do one of the following:
   - **ospfv3 encryption** {**ipsec spi** *spi* **esp** *encryption-algorithm key-encryption-type key authentication-algorithm key-encryption-type key* | **null**}
   - **ipv6 ospf encryption** {**ipsec spi** *spi* **esp** {*encryption-algorithm* [[*key-encryption-type*] *key*] | **null**} *authentication-algorithm* [*key-encryption-type*] *key* | **null**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface ethernet 0/0`	Specifies an interface type and number, and places the device in interface configuration mode.

Command or Action	Purpose

Step 4 Do one of the following:
• **ospfv3 encryption** {**ipsec spi** *spi* **esp** *encryption-algorithm key-encryption-type key authentication-algorithm key-encryption-type key* | **null**}
• **ipv6 ospf encryption** {**ipsec spi** *spi* **esp** {*encryption-algorithm* [[*key-encryption-type*] *key*] | **null**} *authentication-algorithm* [*key-encryption-type*] *key* | **null**}
Example:
`Device(config-if)# ospfv3 encryption ipsec spi 1001 esp null md5 0 27576134094768132473302031209727`
Example:
`Device(config-if) ipv6 ospf encryption ipsec spi 1001 esp null sha1 123456789A123456789B123456789C123456789D`
	Specifies the encryption type for an interface.

Defining Authentication in an OSPFv3 Area

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router ospf** *process-id*
4. **area** *area-id* **authentication ipsec spi** *spi authentication-algorithm* [*key-encryption-type*] *key*

DETAILED STEPS

Command or Action	Purpose
Step 1 **enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2 **configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Command or Action	Purpose
Step 3 **ipv6 router ospf** *process-id* **Example:** `Device(config)# ipv6 router ospf 1`	Enables OSPFv3 router configuration mode.
Step 4 **area** *area-id* **authentication ipsec spi** *spi authentication-algorithm* [*key-encryption-type*] *key* **Example:** `Device(config-rtr)# area 1 authentication ipsec spi 678 md5 1234567890ABCDEF1234567890ABCDEF`	Enables authentication in an OSPFv3 area.

Defining Encryption in an OSPFv3 Area

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router ospf** *process-id*
4. **area** *area-id* **encryption ipsec spi** *spi* **esp** { *encryption-algorithm* [| *key-encryption-type*] *key* | **null**} *authentication-algorithm* [| *key-encryption-type*] *key*

DETAILED STEPS

Command or Action	Purpose
Step 1 **enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2 **configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ipv6 router ospf** *process-id* **Example:** `Device(config)# ipv6 router ospf 1`	Enables OSPFv3 router configuration mode.
Step 4	**area** *area-id* **encryption ipsec spi** *spi* **esp** { *encryption-algorithm* [| *key-encryption-type*] *key* | **null**} *authentication-algorithm* [| *key-encryption-type*] *key* **Example:** `Device(config-rtr)# area 1 encryption ipsec spi 500 esp null md5 1aaa2bbb3ccc4ddd5eee6fff7aaa8bbb`	Enables encryption in an OSPFv3 area.

Defining Authentication and Encryption for a Virtual Link in an OSPFv3 Area

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router ospf** *process-id*
4. **area** *area-id* **virtual-link** *router-id* **authentication ipsec spi** *spi authentication-algorithm* [*key-encryption-type*] *key*
5. **area** *area-id* **virtual-link** *router-id* **encryption ipsec spi** *spi* **esp** {*encryption-algorithm* [*key-encryption-type*] *key* | **null**} *authentication-algorithm* [*key-encryption-type*] *key*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ipv6 router ospf** *process-id* **Example:** `Device(config)# ipv6 router ospf 1`	Enables OSPFv3 router configuration mode.
Step 4	**area** *area-id* **virtual-link** *router-id* **authentication ipsec spi** *spi authentication-algorithm* [*key-encryption-type*] *key* **Example:** `Device(config-rtr)# area 1 virtual-link 10.0.0.1 authentication ipsec spi 940 md5 1234567890ABCDEF1234567890ABCDEF`	Enables authentication for virtual links in an OSPFv3 area.
Step 5	**area** *area-id* **virtual-link** *router-id* **encryption ipsec spi** *spi* **esp** {*encryption-algorithm* [*key-encryption-type*] *key* | **null**} *authentication-algorithm* [*key-encryption-type*] *key* **Example:** `Device(config-rtr)# area 1 virtual-link 10.1.0.1 hello-interval 2 dead-interval 10 encryption ipsec spi 3944 esp null sha1 123456789A123456789B123456789C123456789D`	Enables encryption for virtual links in an OSPFv3 area.

Configuring NBMA Interfaces in OSPFv3

You can customize OSPFv3 in your network to use NBMA interfaces. OSPFv3 cannot automatically detect neighbors over NBMA interfaces. On an NBMA interface, you must configure your neighbors manually using interface configuration mode.

Before you configure NBMA interfaces, you must perform the following tasks:

- Configure your network to be an NBMA network
- Identify each neighbor

Note

- You cannot automatically detect neighbors when using NBMA interfaces. You must manually configure your router to detect neighbors when using an NBMA interface.
- When the **ipv6 ospf neighbor** command is configured, the IPv6 address used must be the link-local address of the neighbor.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **frame-relay map ipv6** *ipv6-address dlci* [**broadcast**] [**cisco**] [**ietf**] [**payload-compression** {**packet-by-packet** | **frf9 stac** [*hardware-options*] | **data-stream stac** [*hardware-options*]}]
5. **ipv6 ospf neighbor** *ipv6-address* [**priority** *number*] [**poll-interval** *seconds*] [**cost** *number*] [**database-filter all out**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface serial 0`	Specifies an interface type and number, and places the device in interface configuration mode.
Step 4	**frame-relay map ipv6** *ipv6-address dlci* [**broadcast**] [**cisco**] [**ietf**] [**payload-compression** {**packet-by-packet**	**frf9 stac** [*hardware-options*]	**data-stream stac** [*hardware-options*]}] **Example:** `Device(config-if)#  frame-relay map ipv6 FE80::A8BB:CCFF:FE00:C01 120`	Defines the mapping between a destination IPv6 address and the data-link connection identifier (DLCI) used to connect to the destination address. • In this example, the NBMA link is Frame Relay. For other kinds of NBMA links, different mapping commands are used.
Step 5	**ipv6 ospf neighbor** *ipv6-address* [**priority** *number*] [**poll-interval** *seconds*] [**cost** *number*] [**database-filter all out**] **Example:** `Device(config-if) ipv6 ospf neighbor FE80::A8BB:CCFF:FE00:C01`	Configures an OSPFv3 neighboring device.

Tuning LSA and SPF Timers for OSPFv3 Fast Convergence

This task can be performed in Cisco IOS Release 15.1(3)S and 15.2(1)T and later releases.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router ospfv3** [*process-id*]
4. **timers lsa arrival** *milliseconds*
5. **timers pacing flood** *milliseconds*
6. **timers pacing lsa-group** *seconds*
7. **timers pacing retransmission** *milliseconds*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router ospfv3** [*process-id*] **Example:** `Device(config)# router ospfv3 1`	Enables OSPFv3 router configuration mode for the IPv4 or IPv6 address family.
Step 4	**timers lsa arrival** *milliseconds* **Example:** `Device(config-rtr)# timers lsa arrival 300`	Sets the minimum interval at which the software accepts the same LSA from OSPFv3 neighbors.
Step 5	**timers pacing flood** *milliseconds* **Example:** `Device(config-rtr)# timers pacing flood 30`	Configures LSA flood packet pacing.

	Command or Action	Purpose
Step 6	**timers pacing lsa-group** *seconds* **Example:** `Device(config-router)# timers pacing lsa-group 300`	Changes the interval at which OSPFv3 LSAs are collected into a group and refreshed, checksummed, or aged.
Step 7	**timers pacing retransmission** *milliseconds* **Example:** `Device(config-router)# timers pacing retransmission 100`	Configures LSA retransmission packet pacing in IPv4 OSPFv3.

Configuring LSA and SPF Throttling for OSPFv3 Fast Convergence

This task can be performed in releases prior to Cisco IOS Release 15.1(3)S and 15.2(1)T.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router ospf** *process-id*
4. **timers throttle spf** *spf-start spf-hold spf-max-wait*
5. **timers throttle lsa** *start-interval hold-interval max-interval*
6. **timers lsa arrival** *milliseconds*
7. **timers pacing flood** *milliseconds*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ipv6 router ospf** *process-id* **Example:** `Device(config)# ipv6 router ospf 1`	Enables OSPFv3 router configuration mode.
Step 4	**timers throttle spf** *spf-start spf-hold spf-max-wait* **Example:** `Device(config-rtr)# timers throttle spf 200 200 200`	Turns on SPF throttling.
Step 5	**timers throttle lsa** *start-interval hold-interval max-interval* **Example:** `Device(config-rtr)# timers throttle lsa 300 300 300`	Sets rate-limiting values for OSPFv3 LSA generation.
Step 6	**timers lsa arrival** *milliseconds* **Example:** `Device(config-rtr)# timers lsa arrival 300`	Sets the minimum interval at which the software accepts the same LSA from OSPFv3 neighbors.
Step 7	**timers pacing flood** *milliseconds* **Example:** `Device(config-rtr)# timers pacing flood 30`	Configures LSA flood packet pacing.

Enabling Event Logging for LSA and SPF Rate Limiting for the IPv6 and IPv4 Address Family

This task can be performed in Cisco IOS Release 15.1(3)S and 15.2(1)T and later releases.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router ospfv3** [*process-id*]
4. **address-family ipv6 unicast**
5. **event-log** [**one-shot** | **pause** | **size** *number-of-events*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router ospfv3** [*process-id*] **Example:** `Device(config)# router ospfv3 1`	Enables OSPFv3 router configuration mode for the IPv4 or IPv6 address family.

Step 4	**address-family ipv6 unicast**
	
	Example:
	or
	
	Example:
	address-family ipv4 unicast
	
	Example:
	`Device(config-router)# address-family ipv6 unicast`
	or
	
	Example:
	`Device(config-router)# address-family ipv4 unicast`
		Enters IPv6 address family configuration mode for OSPFv3.
		or
		Enters IPv4 address family configuration mode for OSPFv3.

Step 5	**event-log** [**one-shot** | **pause** | **size** *number-of-events*] **Example:** `Device(config-router)# event-log`	Enable OSPFv3 event logging in an IPv4 OSPFv3 process.

Enabling Event Logging for LSA and SPF Rate Limiting

This task can be performed in releases prior to Cisco IOS Release 15.1(3)S and 15.2(1)T.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router ospf** *process-id*
4. **event-log** [**size** [*number of events*]] [**one-shot**] [**pause**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router ospf** *process-id* **Example:** `Device(config)# ipv6 router ospf 1`	Enables OSPFv3 router configuration mode.
Step 4	**event-log** [**size** [*number of events*]] [**one-shot**] [**pause**] **Example:** `Device(config-router)# event-log size 10000 one-shot`	Enables event logging.

Clearing the Content of an Event Log

SUMMARY STEPS

1. **enable**
2. **clear ipv6 ospf** [*process-id*] **events**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ipv6 ospf** [*process-id*] **events** **Example:** `Router# clear ipv6 ospf 1 events`	Clears the OSPFv3 event log content based on the OSPFv3 routing process ID.

Calculating OSPFv3 External Path Preferences per RFC 5340

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router ospfv3** [*process-id*]
4. **no compatible rfc1583**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router ospfv3** [*process-id*] **Example:** `Device(config)# router ospfv3 1`	Enables OSPFv3 router configuration mode for the IPv4 or IPv6 address family.

	Command or Action	Purpose
Step 4	**no compatible rfc1583** **Example:** `Device(config-router)# no compatible rfc1583`	Changes the method used to calculate external path preferences per RFC 5340.

Enabling OSPFv3 Graceful Restart

Enabling OSPFv3 Graceful Restart on a Graceful-Restart-Capable Router

This task can be performed in Cisco IOS Release 15.1(3)S and 15.2(1)T and later releases.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router ospfv3** [*process-id*]
4. **graceful-restart** [**restart-interval** *interval*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router ospfv3** [*process-id*] **Example:** `Router(config)# router ospfv3 1`	Enables OSPFv3 router configuration mode for the IPv4 or IPv6 address family.

	Command or Action	Purpose
Step 4	**graceful-restart** [**restart-interval** *interval*] **Example:** `Router(config-rtr)# graceful-restart`	Enables the OSPFv3 graceful restart feature on a graceful-restart-capable router.

- Enabling OSPFv3 Graceful Restart on a Graceful-Restart-Capable Router, page 642

Enabling OSPFv3 Graceful Restart on a Graceful-Restart-Capable Router

The task can be performed in releases prior to Cisco IOS Release 15.1(3)S and 15.2(1)T.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router ospf** *process-id*
4. **graceful-restart** [**restart-interval** *interval*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router ospf** *process-id* **Example:** `Device(config)# ipv6 router ospf 1`	Enables OSPFv3 router configuration mode.
Step 4	**graceful-restart** [**restart-interval** *interval*] **Example:** `Device(config-rtr)# graceful-restart`	Enables the OSPFv3 graceful restart feature on a graceful-restart-capable router.

Enabling OSPFv3 Graceful Restart on a Graceful-Restart-Aware Router

This task can be performed in Cisco IOS Release 15.1(3)S and 15.2(1)T and later releases.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router ospfv3** [*process-id*]
4. **graceful-restart helper** {**disable** | **strict-lsa-checking**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router ospfv3** [*process-id*] **Example:** `Device(config)# router ospfv3 1`	Enables OSPFv3 router configuration mode for the IPv4 or IPv6 address family.
Step 4	**graceful-restart helper** {**disable** | **strict-lsa-checking**} **Example:** `Device(config-rtr)# graceful-restart helper strict-lsa-checking`	Enables the OSPFv3 graceful restart feature on a graceful-restart-aware device.

• Enabling OSPFv3 Graceful Restart on a Graceful-Restart-Aware Router, page 643

Enabling OSPFv3 Graceful Restart on a Graceful-Restart-Aware Router

The task can be performed in releases prior to Cisco IOS Release 15.1(3)S and 15.2(1)T.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router ospf** *process-id*
4. **graceful-restart helper** {**disable** | **strict-lsa-checking**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router ospf** *process-id* **Example:** `Device(config)# ipv6 router ospf 1`	Enables OSPFv3 router configuration mode.
Step 4	**graceful-restart helper** {**disable**	**strict-lsa-checking**} **Example:** `Device(config-rtr)# graceful-restart helper strict-lsa-checking`	Enables the OSPFv3 graceful restart feature on a graceful-restart-aware device.

Forcing an SPF Calculation

SUMMARY STEPS

1. **enable**
2. **clear ospfv3** [*process-id*] **force-spf**
3. **clear ospfv3** [*process-id*] **process**
4. **clear ospfv3** [*process-id*] **redistribution**
5. **clear ipv6 ospf** [*process-id*] {**process** | **force-spf** | **redistribution**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ospfv3** [*process-id*] **force-spf** **Example:** `Device# clear ospfv3 1 force-spf`	Runs SPF calculations for an OSPFv3 process. • If the **clear ospfv3 force-spf** command is configured, it overwrites the **clear ipv6 ospf** configuration. • Once the **clear ospfv3 force-spf** command has been used, the **clear ipv6 ospf** command cannot be used.
Step 3	**clear ospfv3** [*process-id*] **process** **Example:** `Device# clear ospfv3 2 process`	Resets an OSPFv3 process. • If the **clear ospfv3 force-spf** command is configured, it overwrites the **clear ipv6 ospf** configuration. • Once the **clear ospfv3 force-spf** command has been used, the **clear ipv6 ospf** command cannot be used.
Step 4	**clear ospfv3** [*process-id*] **redistribution** **Example:** `Device# clear ospfv3 redistribution`	Clears OSPFv3 route redistribution. • If the **clear ospfv3 force-spf** command is configured, it overwrites the **clear ipv6 ospf** configuration. • Once the **clear ospfv3 force-spf** command has been used, the **clear ipv6 ospf** command cannot be used.
Step 5	**clear ipv6 ospf** [*process-id*] {**process** | **force-spf** | **redistribution**} **Example:** `Device# clear ipv6 ospf force-spf`	Clears the OSPFv3 state based on the OSPFv3 routing process ID, and forces the start of the SPF algorithm. • If the **clear ospfv3 force-spf** command is configured, it overwrites the **clear ipv6 ospf** configuration. • Once the **clear ospfv3 force-spf** command has been used, the **clear ipv6 ospf** command cannot be used.

Verifying OSPFv3 Configuration and Operation

This task is optional. The commands in this task are available in Cisco IOS Release 15.1(3)S and 15.2(1)T and later releases.

SUMMARY STEPS

1. **enable**
2. **show ospfv3** [*process-id*] [*address-family*] **border-routers**
3. **show ospfv3** [*process-id* [*area-id*]] [*address-family*] **database** [**database-summary** | **internal** | **external** [*ipv6-prefix*] [*link-state-id*] | **grace** | **inter-area prefix** [*ipv6-prefix* | *link-state-id*] | **inter-area router** [*destination-router-id* | *link-state-id*] | **link** [**interface** *interface-name* | *link-state-id*] | **network** [*link-state-id*] | **nssa-external** [*ipv6-prefix*] [*link-state-id*] | **prefix** [**ref-lsa** {**router** | **network**} | *link-state-id*] | **promiscuous** | **router** [*link-state-id*] | **unknown** [{**area** | **as** | **link**} [*link-state-id*]] [**adv-router** *router-id*] [**self-originate**]
4. **show ospfv3** [*process-id*] [*address-family*] **events** [**generic** | **interface** | **lsa** | **neighbor** | **reverse** | **rib** | **spf**]
5. **show ospfv3** [*process-id*] [*area-id*] [*address-family*] **flood-list** *interface-type interface-number*
6. **show ospfv3** [*process-id*] [*address-family*] **graceful-restart**
7. **show ospfv3** [*process-id*] [*area-id*] [*address-family*] **interface** [*type number*] [**brief**]
8. **show ospfv3** [*process-id*] [*area-id*] [*address-family*] **neighbor** [*interface-type interface-number*] [*neighbor-id*] [**detail**]
9. **show ospfv3** [*process-id*] [*area-id*] [*address-family*] **request-list**[*neighbor*] [*interface*] [*interface-neighbor*]
10. **show ospfv3** [*process-id*] [*area-id*] [*address-family*] **retransmission-list** [*neighbor*] [*interface*] [*interface-neighbor*]
11. **show ospfv3** [*process-id*] [*address-family*] **statistic** [**detail**]
12. **show ospfv3** [*process-id*] [*address-family*] **summary-prefix**
13. **show ospfv3** [*process-id*] [*address-family*] **timers rate-limit**
14. **show ospfv3** [*process-id*] [*address-family*] **traffic**[*interface-type interface-number*]
15. **show ospfv3** [*process-id*] [*address-family*] **virtual-links**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ospfv3** [*process-id*] [*address-family*] **border-routers** **Example:** `Device# show ospfv3 border-routers`	Displays the internal OSPFv3 routing table entries to an ABR and ASBR.

	Command or Action	Purpose
Step 3	**show ospfv3** [*process-id* [*area-id*]] [*address-family*] **database** [**database-summary** | **internal** | **external** [*ipv6-prefix*] [*link-state-id*] | **grace** | **inter-area prefix** [*ipv6-prefix* | *link-state-id*] | **inter-area router** [*destination-router-id* | *link-state-id*] | **link** [**interface** *interface-name* | *link-state-id*] | **network** [*link-state-id*] | **nssa-external** [*ipv6-prefix*] [*link-state-id*] | **prefix** [**ref-lsa** {**router** | **network**} | *link-state-id*] | **promiscuous** | **router** [*link-state-id*] | **unknown** [{**area** | **as** | **link**} [*link-state-id*]] [**adv-router** *router-id*] [**self-originate**] **Example:** `Device# show ospfv3 database`	Displays lists of information related to the OSPFv3 database for a specific device.
Step 4	**show ospfv3** [*process-id*] [*address-family*] **events** [**generic** | **interface** | **lsa** | **neighbor** | **reverse** | **rib** | **spf**] **Example:** `Device# show ospfv3 events`	Displays detailed information about OSPFv3 events.
Step 5	**show ospfv3** [*process-id*] [*area-id*] [*address-family*] **flood-list** *interface-type interface-number* **Example:** `Device# show ospfv3 flood-list`	Displays a list of OSPFv3 LSAs waiting to be flooded over an interface.
Step 6	**show ospfv3** [*process-id*] [*address-family*] **graceful-restart** **Example:** `Device# show ospfv3 graceful-restart`	Displays OSPFv3 graceful restart information.
Step 7	**show ospfv3** [*process-id*] [*area-id*] [*address-family*] **interface** [*type number*] [**brief**] **Example:** `Device# show ospfv3 interface`	Displays OSPFv3-related interface information.
Step 8	**show ospfv3** [*process-id*] [*area-id*] [*address-family*] **neighbor** [*interface-type interface-number*] [*neighbor-id*] [**detail**] **Example:** `Device# show ospfv3 neighbor`	Displays OSPFv3 neighbor information on a per-interface basis.

	Command or Action	Purpose
Step 9	**show ospfv3** [*process-id*] [*area-id*] [*address-family*] **request-list**[*neighbor*] [*interface*] [*interface-neighbor*] **Example:** `Device# show ospfv3 request-list`	Displays a list of all LSAs requested by a device.
Step 10	**show ospfv3** [*process-id*] [*area-id*] [*address-family*] **retransmission-list** [*neighbor*] [*interface*] [*interface-neighbor*] **Example:** `Device# show ospfv3 retransmission-list`	Displays a list of all LSAs waiting to be re-sent.
Step 11	**show ospfv3** [*process-id*] [*address-family*] **statistic** [**detail**] **Example:** `Device# show ospfv3 statistics`	Displays OSPFv3 SPF calculation statistics.
Step 12	**show ospfv3** [*process-id*] [*address-family*] **summary-prefix** **Example:** `Device# show ospfv3 summary-prefix`	Displays a list of all summary address redistribution information configured under an OSPFv3 process.
Step 13	**show ospfv3** [*process-id*] [*address-family*] **timers rate-limit** **Example:** `Device# show ospfv3 timers rate-limit`	Displays all of the LSAs in the rate limit queue.
Step 14	**show ospfv3** [*process-id*] [*address-family*] **traffic**[*interface-type interface-number*] **Example:** `Device# show ospfv3 traffic`	Displays OSPFv3 traffic statistics.
Step 15	**show ospfv3** [*process-id*] [*address-family*] **virtual-links** **Example:** `Device# show ospfv3 virtual-links`	Displays parameters and the current state of OSPFv3 virtual links.

- Verifying OSPFv3 Configuration and Operation, page 649

- Examples, page 650

Verifying OSPFv3 Configuration and Operation

SUMMARY STEPS

1. **enable**
2. **show ipv6 ospf** [*process-id*] [*area-id*] **interface** [*interface-type interface-number*]
3. **show ipv6 ospf** [*process-id*] [*area-id*]
4. **show crypto ipsec policy** [**name** *policy-name*]
5. **show crypto ipsec sa** [**map** *map-name* | **address** | **identity** | **interface** *type number* | **peer** [**vrf** *fvrf-name*] **address** | **vrf** *ivrf-name* | **ipv6** [*interface-type interface-number*]] [**detail**]
6. **show ipv6 ospf** [*process-id*] **event** [**generic** | **interface** | **lsa** | **neighbor** | **reverse** | **rib** | **spf**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ipv6 ospf** [*process-id*] [*area-id*] **interface** [*interface-type interface-number*] **Example:** `Device# show ipv6 ospf interface`	Displays OSPFv3-related interface information.
Step 3	**show ipv6 ospf** [*process-id*] [*area-id*] **Example:** `Device# show ipv6 ospf`	Displays general information about OSPFv3 routing processes.
Step 4	**show crypto ipsec policy** [**name** *policy-name*] **Example:** `Device# show crypto ipsec policy`	Displays the parameters for each IPsec parameter.

Command or Action	Purpose
Step 5 **show crypto ipsec sa** [**map** *map-name* | **address** | **identity** | **interface** *type number* | **peer** [**vrf** *fvrf-name*] **address** | **vrf** *ivrf-name* | **ipv6** [*interface-type interface-number*]] [**detail**] **Example:** `Device# show crypto ipsec sa ipv6`	Displays the settings used by current security associations (SAs).
Step 6 **show ipv6 ospf** [*process-id*] **event** [**generic** | **interface** | **lsa** | **neighbor** | **reverse** | **rib** | **spf**] **Example:** `Device# show ipv6 ospf event spf`	Displays detailed information about OSPFv3 events.

Examples

Sample Output from the show ipv6 ospf interface Command

The following is sample output from the **show ipv6 ospf interface** command with regular interfaces and a virtual link that are protected by encryption and authentication:

```
Device# show ipv6 ospf interface

OSPFv3_VL1 is up, line protocol is up
   Interface ID 69
   Area 0, Process ID 1, Instance ID 0, Router ID 10.0.0.1
   Network Type VIRTUAL_LINK, Cost: 64
   Configured as demand circuit.
   Run as demand circuit.
   DoNotAge LSA allowed.
   NULL encryption SHA-1 auth SPI 3944, secure socket UP (errors: 0)
   Transmit Delay is 1 sec, State POINT_TO_POINT,
   Timer intervals configured, Hello 2, Dead 10, Wait 40, Retransmit 5
     Hello due in 00:00:00
   Index 1/3/5, flood queue length 0
   Next 0x0(0)/0x0(0)/0x0(0)
   Last flood scan length is 1, maximum is 1
   Last flood scan time is 0 msec, maximum is 0 msec
   Neighbor Count is 1, Adjacent neighbor count is 1
     Adjacent with neighbor 10.2.0.1  (Hello suppressed)
   Suppress hello for 1 neighbor(s)
OSPFv3_VL0 is up, line protocol is up
   Interface ID 67
   Area 0, Process ID 1, Instance ID 0, Router ID 10.0.0.1
   Network Type VIRTUAL_LINK, Cost: 128
   Configured as demand circuit.
   Run as demand circuit.
   DoNotAge LSA allowed.
   MD5 authentication SPI 940, secure socket UP (errors: 0)
   Transmit Delay is 1 sec, State POINT_TO_POINT,
   Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
     Hello due in 00:00:09
   Index 1/2/4, flood queue length 0
   Next 0x0(0)/0x0(0)/0x0(0)
   Last flood scan length is 1, maximum is 10
   Last flood scan time is 0 msec, maximum is 0 msec
```

```
   Neighbor Count is 1, Adjacent neighbor count is 1
     Adjacent with neighbor 10.1.0.1  (Hello suppressed)
   Suppress hello for 1 neighbor(s)
Ethernet1/0 is up, line protocol is up
   Link Local Address FE80::A8BB:CCFF:FE00:6601, Interface ID 6
   Area 0, Process ID 1, Instance ID 0, Router ID 10.0.0.1
   Network Type BROADCAST, Cost: 10
   Transmit Delay is 1 sec, State DR, Priority 1
   Designated Router (ID) 10.0.0.1, local address FE80::A8BB:CCFF:FE00:6601
   No backup designated router on this network
   Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
     Hello due in 00:00:09
   Index 1/1/1, flood queue length 0
   Next 0x0(0)/0x0(0)/0x0(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)
Serial12/0 is up, line protocol is up
   Link Local Address FE80::A8BB:CCFF:FE00:6600, Interface ID 50
   Area 1, Process ID 1, Instance ID 0, Router ID 10.0.0.1
   Network Type POINT_TO_POINT, Cost: 64
   AES-CBC encryption SHA-1 auth SPI 2503, secure socket UP (errors: 0)
   authentication NULL
   Transmit Delay is 1 sec, State POINT_TO_POINT,
   Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
     Hello due in 00:00:09
   Index 1/2/3, flood queue length 0
   Next 0x0(0)/0x0(0)/0x0(0)
   Last flood scan length is 1, maximum is 5
   Last flood scan time is 0 msec, maximum is 0 msec
   Neighbor Count is 1, Adjacent neighbor count is 1
     Adjacent with neighbor 10.2.0.1
   Suppress hello for 0 neighbor(s)
Serial11/0 is up, line protocol is up
   Link Local Address FE80::A8BB:CCFF:FE00:6600, Interface ID 46
   Area 1, Process ID 1, Instance ID 0, Router ID 10.0.0.1
   Network Type POINT_TO_POINT, Cost: 64
   MD5 authentication (Area) SPI 500, secure socket UP (errors: 0)
   Transmit Delay is 1 sec, State POINT_TO_POINT,
   Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
     Hello due in 00:00:09
   Index 1/1/2, flood queue length 0
   Next 0x0(0)/0x0(0)/0x0(0)
   Last flood scan length is 1, maximum is 5
   Last flood scan time is 0 msec, maximum is 0 msec
   Neighbor Count is 1, Adjacent neighbor count is 1
     Adjacent with neighbor 10.0.0.1
   Suppress hello for 0 neighbor(s)
```

Sample Output from the show ipv6 ospf Command

The following is sample output from the **show ipv6 ospf** command:

```
Device# show ipv6 ospf

Routing Process "ospfv3 1" with ID 172.16.3.3
 It is an autonomous system boundary router
 Redistributing External Routes from,
    static
 SPF schedule delay 5 secs, Hold time between two SPFs 10 secs
 Minimum LSA interval 5 secs. Minimum LSA arrival 1 secs
 LSA group pacing timer 240 secs
 Interface flood pacing timer 33 msecs
 Retransmission pacing timer 66 msecs
 Number of external LSA 1. Checksum Sum 0x218D
 Number of areas in this router is 1. 1 normal 0 stub 0 nssa
    Area 1
        Number of interfaces in this area is 2
        SPF algorithm executed 9 times
        Number of LSA 15. Checksum Sum 0x67581
        Number of DCbitless LSA 0
```

```
        Number of indication LSA 0
        Number of DoNotAge LSA 0
        Flood list length 0
```

Sample Output from the show crypto ipsec policy Command

The following is sample output from the **show crypto ipsec policy** command:

```
Device# show crypto ipsec policy

Crypto IPsec client security policy data
Policy name:     OSPFv3-1-1000
Policy refcount: 1
Inbound  AH SPI: 1000 (0x3E8)
Outbound AH SPI: 1000 (0x3E8)
Inbound  AH Key: 1234567890ABCDEF1234567890ABCDEF
Outbound AH Key: 1234567890ABCDEF1234567890ABCDEF
Transform set:   ah-md5-hmac
```

Sample Output from the show crypto ipsec sa ipv6 Command

The following is sample output from the **show crypto ipsec sa ipv6** command:

```
Device# show crypto ipsec sa ipv6

IPv6 IPsec SA info for interface Ethernet0/0
   protected policy name:OSPFv3-1-1000
   IPsec created ACL name:Ethernet0/0-ipsecv6-ACL
   local  ident (addr/prefixlen/proto/port):(FE80::/10/89/0)
   remote ident (addr/prefixlen/proto/port):(::/0/89/0)
   current_peer:::
     PERMIT, flags={origin_is_acl,}
    #pkts encaps:21, #pkts encrypt:0, #pkts digest:21
    #pkts decaps:20, #pkts decrypt:0, #pkts verify:20
    #pkts compressed:0, #pkts decompressed:0
    #pkts not compressed:0, #pkts compr. failed:0
    #pkts not decompressed:0, #pkts decompress failed:0
    #send errors 0, #recv errors 0
    local crypto endpt. ::, remote crypto endpt. ::
    path mtu 1500, media mtu 1500
    current outbound spi:0x3E8(1000)
     inbound ESP SAs:
     inbound AH SAs:
      spi:0x3E8(1000)
        transform:ah-md5-hmac ,
        in use settings ={Transport, }
        slot:0, conn_id:2000, flow_id:1, crypto map:N/R
        no sa timing (manual-keyed)
        replay detection support:N
     inbound PCP SAs:
     outbound ESP SAs:
     outbound AH SAs:
      spi:0x3E8(1000)
        transform:ah-md5-hmac ,
        in use settings ={Transport, }
        slot:0, conn_id:2001, flow_id:2, crypto map:N/R
        no sa timing (manual-keyed)
        replay detection support:N
     outbound PCP SAs:
```

Sample Output from the show ipv6 ospf graceful-restart Command

The following is sample output from the **show ipv6 ospf graceful-restart** command:

```
Device# show ipv6 ospf graceful-restart

Routing Process "ospf 1"
 Graceful Restart enabled
```

```
  restart-interval limit: 120 sec, last restart 00:00:15 ago (took 36 secs)
 Graceful Restart helper support enabled
 Router status : Active
 Router is running in SSO mode
 OSPF restart state : NO_RESTART
 Router ID 10.1.1.1, checkpoint Router ID 10.0.0.0
```

Configuration Examples for Implementing OSPFv3

Example: Enabling OSPFv3 on an Interface Configuration

The following example shows the command to use to configure OSPFv3 routing process 109 to run on the interface and puts it in area 1:

```
ipv6 ospf 109 area 1
```

Example: Defining an OSPFv3 Area Range

The following example shows how to specify an OSPFv3 area range:

```
interface Ethernet7/0
 ipv6 address 2001:DB8:0:7::/64 eui-64
 ipv6 enable
 ipv6 ospf 1 area 1
!
interface Ethernet8/0
 ipv6 address 2001:DB8:0:8::/64 eui-64
 ipv6 enable
 ipv6 ospf 1 area 1
!
interface Ethernet9/0
 ipv6 address 2001:DB8:0:9::/64 eui-64
 ipv6 enable
 ipv6 ospf 1 area 1
!
ipv6 router ospf 1
 router-id 10.11.11.1
 area 1 range 2001:DB8::/48
```

Example: Defining Authentication on an Interface

The following example shows how to define authentication on Ethernet interface 0/0:

```
interface Ethernet0/0
 ipv6 enable
 ipv6 ospf 1 area 0
 ipv6 ospf authentication ipsec spi 500 md5 1234567890ABCDEF1234567890ABCDEF
interface Ethernet0/0
 ipv6 enable
```

```
 ipv6 ospf authentication null
 ipv6 ospf 1 area 0
```

Example: Defining Authentication in an OSPFv3 Area

The following example shows how to define authentication on OSPFv3 area 0:

```
ipv6 router ospf 1
 router-id 10.11.11.1
 area 0 authentication ipsec spi 1000 md5 1234567890ABCDEF1234567890ABCDEF
```

Example: Configuring NBMA Interfaces

The following example shows how to configure an OSPFv3 neighboring router with the IPv6 address of FE80::A8BB:CCFF:FE00:C01.

```
interface serial 0
 ipv6 enable
 ipv6 ospf 1 area 0
 encapsulation frame-relay
 frame-relay map ipv6 FE80::A8BB:CCFF:FE00:C01 120
 ipv6 ospf neighbor FE80::A8BB:CCFF:FE00:C0
```

Example: Configuring LSA and SPF Throttling for OSPFv3 Fast Convergence

The following example show how to display the configuration values for SPF and LSA throttling timers:

```
Router# show ipv6 ospf

Routing Process "ospfv3 1" with ID 10.9.4.1
Event-log enabled, Maximum number of events: 1000, Mode: cyclic
 It is an autonomous system boundary router
 Redistributing External Routes from,
    ospf 2
 Initial SPF schedule delay 5000 msecs
 Minimum hold time between two consecutive SPFs 10000 msecs
 Maximum wait time between two consecutive SPFs 10000 msecs
 Minimum LSA interval 5 secs
 Minimum LSA arrival 1000 msecs
```

Example: Forcing SPF Configuration

The following example shows how to trigger SPF to redo the SPF and repopulate the routing tables:

```
clear ipv6 ospf force-spf
```

Additional References

Related Documents

Related Topic	Document Title
Configuring a router ID in OSPF	• "Configuring OSPF" in *Cisco IOS IP Routing Protocols Configuration Guide* • *Cisco IOS IP Routing Protocols Command Reference*
OSPFv3 commands	*Cisco IOS IPv6 Command Reference*
IPv6 supported feature list	"Start Here: Cisco IOS Software Release Specifics for IPv6 Features" in the *Cisco IOS IPv6 Configuration Guide*
Implementing basic IPv6 connectivity	"Implementing IPv6 Addressing and Basic Connectivity" in the *Cisco IOS IPv6 Configuration Guide*
IPsec for IPv6	"Implementing IPsec for IPv6 Security" in the *Cisco IOS IPv6 Configuration Guide*
BFD support for OSPFv3	"Implementing Bidirectional Forwarding Detection for IPv6" in the *Cisco IOS IPv6 Configuration Guide*
Stateful switchover	"Stateful Switchover" in the *Cisco IOS High Availability Configuration Guide*
Cisco nonstop forwarding	"Cisco Nonstop Forwarding" in the *Cisco IOS High Availability Configuration Guide*

MIBs

MIBs	MIBs Link
• CISCO-IETF-IP-FORWARD-MIB • CISCO-IETF-IP-MIB	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 1583	*OSPF version 2*
RFC 2401	*Security Architecture for the Internet Protocol*
RFC 2402	*IP Authentication Header*
RFC 2406	*IP Encapsulating Security Payload (ESP)*

RFCs	Title
RFC 3137	*OSPF Stub Router Advertisement*
RFC 4552	*Authentication/Confidentiality for OSPFv3*
RFC 5187	*OSPFv3 Graceful Restart*
RFC 5340	*OSPF for IPv6*
RFC 5838	*Support of Address Families in OSPFv3*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	

Feature Information for Implementing OSPFv3

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 29 Feature Information for Implementing OSPFv3

Feature Name	Releases	Feature Information
IPv6 Routing—Fast Convergence—LSA and SPF Throttling	12.2(33)SB 12.2(33)SRC 12.2(33)XNE 15.0(1)M 15.0(1)SY	The OSPFv3 LSA and SPF Throttling feature provides a dynamic mechanism to slow down link-state advertisement updates in OSPFv3 during times of network instability.

Feature Name	Releases	Feature Information
IPv6 Routing—Force SPF in OSPFv3	12.0(24)S 12.2(18)S 12.2(15)T 12.2(28)SB 12.3 12.3(2)T 12.4 12.4(2)T	This feature enables the OSPFv3 database to be cleared and repopulated before the SPF algorithm is performed.
IPv6 Routing—Load Balancing in OSPFv3	12.0(24)S 12.2(18)S 12.2(15)T 12.2(28)SB 12.3 12.3(2)T 12.4 12.4(2)T	OSPFv3 performs load balancing automatically.
IPv6 Routing—LSA Types in OSPFv3	12.0(24)S 12.2(18)S 12.2(15)T 12.2(28)SB 12.3 12.3(2)T 12.4 12.4(2)T	A router's collection of LSA data is stored in a link-state database. The contents of the database, when subjected to the Dijkstra algorithm, result in the creation of the OSPFv3 routing table.
IPv6 Routing—NBMA Interfaces in OSPFv3	12.0(24)S 12.2(18)S 12.2(15)T 12.2(28)SB 12.3 12.3(2)T 12.4 12.4(2)T	On NBMA networks, the DR or backup DR performs the LSA flooding.

Feature Name	Releases	Feature Information
IPv6 Routing—OSPF for IPv6 (OSPFv3)	12.0(24)S 12.2(18)S 12.2(15)T 12.2(25)SG 12.2(28)SB 12.2(33)SRA 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)M 15.0(1)S	OSPF version 3 for IPv6 expands on OSPF version 2 to provide support for IPv6 routing prefixes and the larger size of IPv6 addresses.
IPv6 Routing—OSPF for IPv6 Authentication Support with IPsec	12.3(4)T 12.4 12.4(2)T 15.0(1)SY1 15.2(1)S	OSPF for IPv6 uses the IPsec secure socket API to add authentication to OSPFv3 packets.
IPv6 Routing—OSPF IPv6 (OSPFv3) IPsec ESP Encryption and Authentication	12.4(9)T 15.0(1)SY1 15.1(1)SG	IPv6 ESP extension headers can be used to provide authentication and confidentiality to OSPFv3.
OSPFv3 Address Families	15.1(3)S 15.2(1)T	The OSPFv3 address families feature enables IPv4 and IPv6 unicast traffic to be supported with a single network topology.
OSPFv3 Dynamic Interface Cost Support	12.4(15)T	OSPFv3 dynamic interface cost support provides enhancements to the OSPFv3 cost metric for supporting mobile ad hoc networking.
OSPFv3 External Path Preference Option	15.1(3)S 15.2(1)T 15.2(3)T	This feature provides a way to calculate external path preferences per RFC 5340.

Feature Name	Releases	Feature Information
OSPFv3 for BFD	12.2(33)SRE 15.0(1)S 15.0(1)SY 15.1(2)T 15.1(1)SG	BFD supports the dynamic routing protocol OSPFv3.
OSPFv3 Graceful Restart	12.2(33)SRE 12.2(33)XNE 12.2(58)SE 15.0(1)M 15.0(1)SY 15.1(1)SG	The Graceful Restart feature in OSPFv3 allows nonstop data forwarding along routes that are already known while the OSPFv3 routing protocol information is being restored.
OSPFv3 Max-Metric Router LSA	15.1(3)S 15.2(1)T 15.2(3)T	The OSPFv3 Max-Metric Router LSA feature enables OSPF to advertise its locally generated router LSAs with a maximum metric.

Implementing IPv6 over MPLS

Multiprotocol Label Switching (MPLS) is deployed by many service providers in their IPv4 networks. Service providers want to introduce IPv6 services to their customers, but changes to their existing IPv4 infrastructure can be expensive and the cost benefit for a small amount of IPv6 traffic does not make economic sense. Several integration scenarios have been developed to leverage an existing IPv4 MPLS infrastructure and add IPv6 services without requiring any changes to the network backbone. This document describes how to implement IPv6 over MPLS.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Implementing IPv6 over MPLS

- This module assumes that you are familiar with IPv4. Refer to the publications referenced in the Prerequisites for Implementing IPv6 over MPLS, page 661 section for IPv4 configuration and command reference information.
- Before the IPv6 Provider Edge Router over MPLS (6PE) feature can be implemented, MPLS must be running over the core IPv4 network. If Cisco routers are used, Cisco Express Forwarding or distributed Cisco Express Forwarding must be enabled for both IPv4 and IPv6 protocols. This module assumes that you are familiar with MPLS.

Information About Implementing IPv6 over MPLS

Benefits of Deploying IPv6 over MPLS Backbones

IPv6 over MPLS backbones enables isolated IPv6 domains to communicate with each other over an MPLS IPv4 core network. This implementation requires only a few backbone infrastructure upgrades and no reconfiguration of core routers because forwarding is based on labels rather than the IP header itself, providing a very cost-effective strategy for the deployment of IPv6.

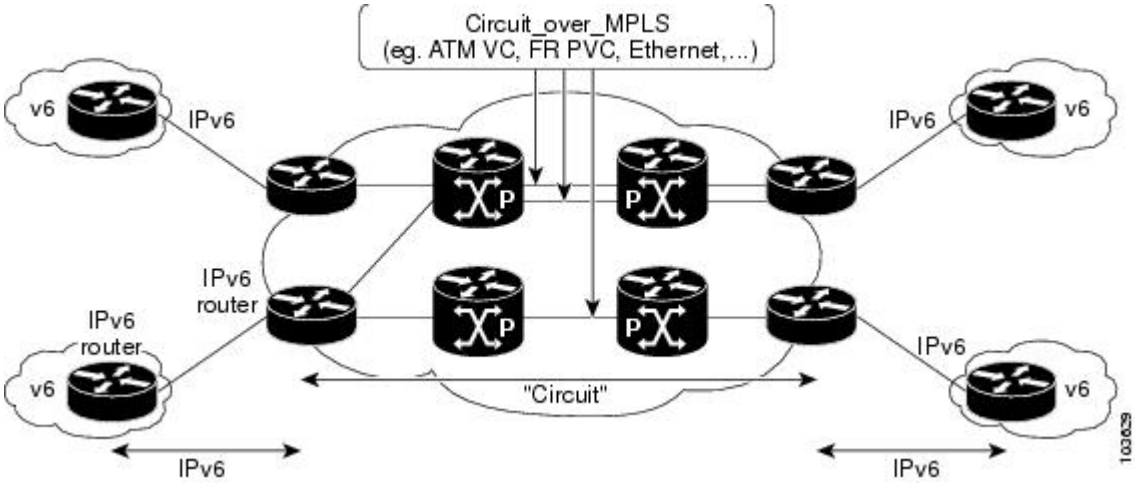
Additionally, the inherent Virtual Private Network (VPN) and MPLS traffic engineering (MPLS-TE) services available within an MPLS environment allow IPv6 networks to be combined into IPv4 VPNs or extranets over an infrastructure supporting IPv4 VPNs and MPLS-TE.

IPv6 over a Circuit Transport over MPLS

Using any circuit transport for deploying IPv6 over MPLS networks has no impact on the operation or infrastructure of MPLS, and requires no configuration changes to the core or provider edge routers. Communication between the remote IPv6 domains runs native IPv6 protocols over a dedicated link, where the underlying mechanisms are fully transparent to IPv6. The IPv6 traffic is tunneled using the Any Transport over MPLS (MPLS/AToM) or Ethernet over MPLS (EoMPLS) feature with the routers connected through an ATM OC-3 or Ethernet interface, respectively.

The figure below shows the configuration for IPv6 over any circuit transport over MPLS.

Figure 43 IPv6 over a Circuit Transport over MPLS

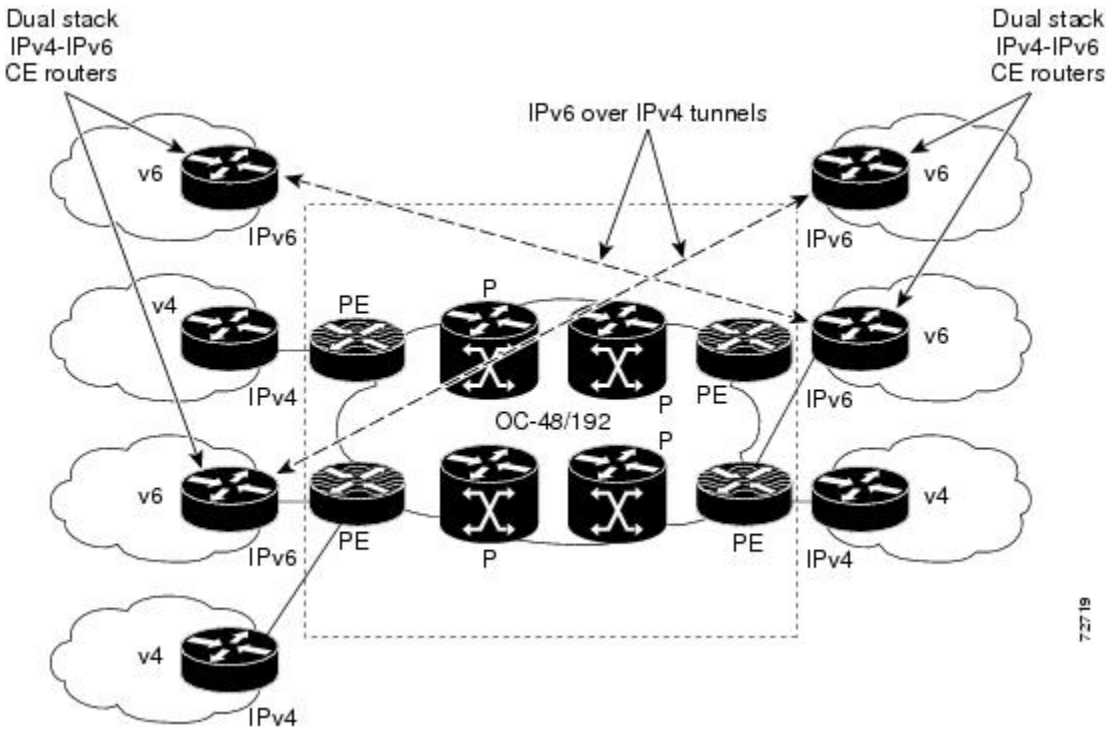


IPv6 Using Tunnels on the Customer Edge Routers

Using tunnels on the customer edge (CE) routers is the simplest way of deploying IPv6 over MPLS networks with no impact on the operation or infrastructure of MPLS, and no configuration changes to the

core or provider edge routers. Communication between the remote IPv6 domains uses standard tunneling mechanisms and requires the CE routers to be configured to run dual IPv4 and IPv6 protocol stacks. The figure below shows the configuration using tunnels on the CE routers.

Figure 44 IPv6 Using Tunnels on the CE Routers



Refer to Implementing Tunneling for IPv6 for configuration information on manually configured tunnels, automatic tunnels, and 6to4 tunnels.

Limitations on using tunnels involve the manual configuring of a mesh of tunnels on the CE routers, creating scaling issues for large networks.

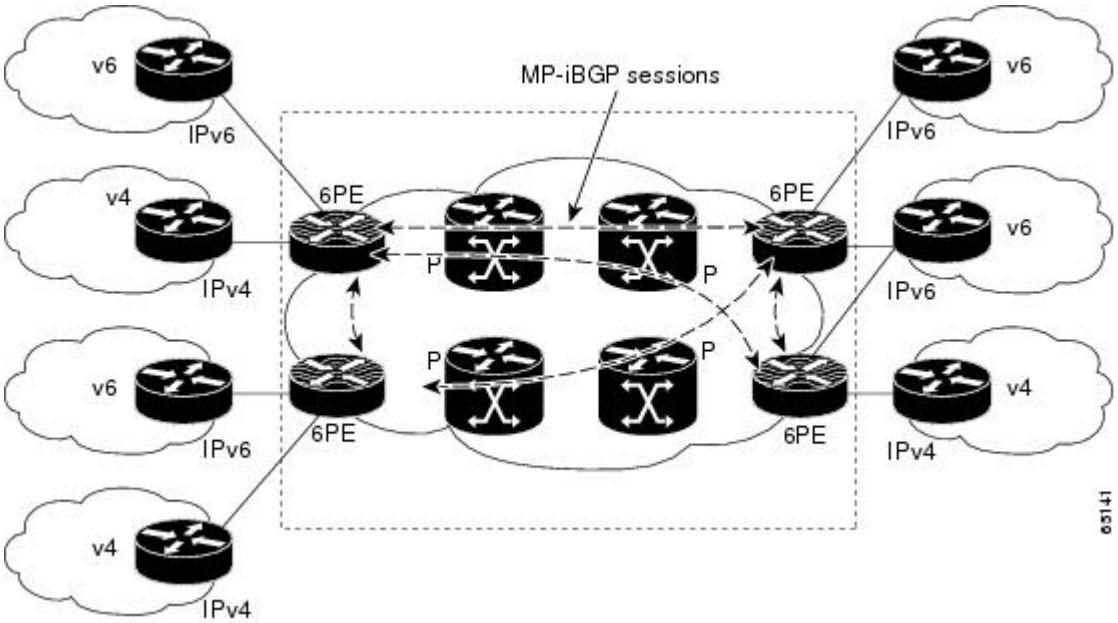
IPv6 on the Provider Edge Routers

The Cisco implementation of IPv6 provider edge router over MPLS is called 6PE, and it enables IPv6 sites to communicate with each other over an MPLS IPv4 core network using MPLS label switched paths (LSPs). This feature relies on multiprotocol Border Gateway Protocol (BGP) extensions in the IPv4 network configuration on the provider edge (PE) router to exchange IPv6 reachability information in addition to an MPLS label for each IPv6 address prefix to be advertised. Edge routers are configured to be dual stack running both IPv4 and IPv6, and use the IPv4 mapped IPv6 address for IPv6 prefix reachability exchange.

A hierarchy of labels is imposed on the 6PE ingress router to keep the IPv6 traffic transparent to all the core routers. The top label provides connectivity inside the IPv4 MPLS core network and the label is distributed by Label Distribution Protocol (LDP), Tag Distribution Protocol (TDP), or Resource Reservation Protocol (RSVP). TDP and LDP can both be used for label distribution, but RSVP is used only in the context of MPLS-TE label exchange. The bottom label, automatically assigned to the IPv6 prefix of the destination, is distributed by multiprotocol BGP and used at each 6PE egress router for IPv6 forwarding.

In the figure below the 6PE routers are configured as dual stack routers able to route both IPv4 and IPv6 traffic. Each 6PE router is configured to run LDP, TDP, or RSVP (if traffic engineering is configured) to bind the IPv4 labels. The 6PE routers use multiprotocol BGP to exchange reachability information with the other 6PE devices within the MPLS domain, and to distribute aggregate IPv6 labels between them. All 6PE and core routers--P routers in Figure 3--within the MPLS domain share a common IPv4 Interior Gateway Protocol (IGP) such as Open Shortest Path First (OSPF) or Integrated Intermediate System-to-Intermediate System (IS-IS).

Figure 45 6PE Router Topology



The interfaces on the 6PE routers connecting to the CE router can be configured to forward IPv6 traffic, IPv4 traffic, or both types of traffic depending on the customer requirements. 6PE routers advertise IPv6 reachability information learned from their 6PE peers over the MPLS cloud. Service providers can delegate an IPv6 prefix from their registered IPv6 prefixes over the 6PE infrastructure; otherwise, there is no impact on the CE router.

The P routers in the core of the network are not aware that they are switching IPv6 packets. Core routers are configured to support MPLS and the same IPv4 IGP as the PE routers to establish internal reachability inside the MPLS cloud. Core routers also use LDP, TDP, or RSVP for binding IPv4 labels. Implementing the Cisco 6PE feature does not have any impact on the MPLS core devices.

Within the MPLS network, IPv6 traffic is forwarded using label switching, making the IPv6 traffic transparent to the core of the MPLS network. No IPv6 over IPv4 tunnels or Layer 2 encapsulation methods are required.

- 6PE Multipath, page 664

6PE Multipath

Internal and external BGP multipath for IPv6 allows the IPv6 device to load balance between several paths (for example, the same neighboring autonomous system or subautonomous system, or the same metric) to reach its destination. The 6PE multipath feature uses MP-iBGP to distribute IPv6 routes over the MPLS IPv4 core network and to attach an MPLS label to each route.

When MP-iBGP multipath is enabled on the 6PE device, all labeled paths are installed in the forwarding table with MPLS information (label stack) when MPLS information is available. This functionality enables 6PE to perform load balancing.

How to Implement IPv6 over MPLS

Deploying IPv6 over a Circuit Transport over MPLS

To deploy IPv6 over a circuit transport over MPLS, the IPv6 routers must be configured for IPv6 connectivity. The MPLS router configuration requires AToM configuration or EoMPLS configuration.

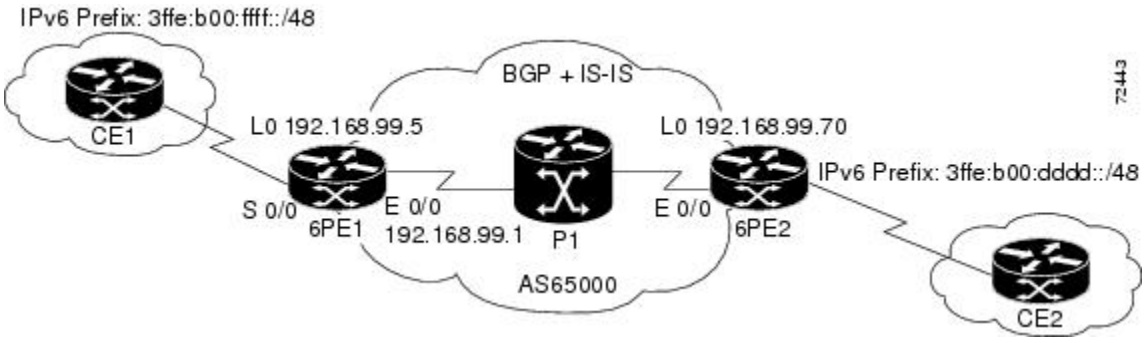
Deploying IPv6 on the Provider Edge Routers (6PE)

Specifying the Source Address Interface on a 6PE Router

Two configuration tasks using the network shown in the figure below are required at the 6PE1 router to enable the 6PE feature.

The customer edge router--CE1 in the figure below--is configured to forward its IPv6 traffic to the 6PE1 router. The P1 router in the core of the network is assumed to be running MPLS, a label distribution protocol, an IPv4 IGP, and Cisco Express Forwarding or distributed Cisco Express Forwarding, and does not require any new configuration to enable the 6PE feature.

Figure 46 6PE Configuration Example



- The 6PE routers--the 6PE1 and 6PE2 routers in the figure below--must be members of the core IPv4 network. The 6PE router interfaces attached to the core network must be running MPLS, the same label distribution protocol, and the same IPv4 IGP, as in the core network.
- The 6PE routers must also be configured to be dual stack to run both IPv4 and IPv6.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **ipv6 cef**
5. **interface** *type number*
6. **ipv6 address** *ipv6-address* / *prefix-length* | *prefix-name sub-bits* / *prefix-length*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Router(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.
Step 4	**ipv6 cef** **Example:** `Router(config)# ipv6 cef`	Enables IPv6 Cisco Express Forwarding.
Step 5	**interface** *type number* **Example:** `Router(config)# interface Serial 0/0`	Specifies an interface type and number and enters interface configuration mode. • In the context of this feature, the interface to be configured is the interface communicating with the CE router.

	Command or Action	Purpose
Step 6	**ipv6 address** *ipv6-address / prefix-length | prefix-name sub-bits / prefix-length* **Example:** `Router(config-if)# ipv6 address 2001:DB8:FFFF::2/64`	Configures an IPv6 address based on an IPv6 general prefix and enable IPv6 processing on an interface.

Binding and Advertising the 6PE Label to Advertise Prefixes

Perform this task to enable the binding and advertising of aggregate labels when advertising IPv6 prefixes to a specified BGP neighbor.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **no bgp default ipv4-unicast**
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
6. **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
7. **address-family ipv6** [**unicast**]
8. **neighbor** {*ip-address* | *peer-group-name*| *ipv6-address*} **activate**
9. **neighbor** {*ip-address* | *ipv6-address*} **send-label**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

Command or Action	Purpose
Step 3 **router bgp** *as-number* **Example:** `Router(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.
Step 4 **no bgp default ipv4-unicast** **Example:** `Router(config-router)# no bgp default ipv4-unicast`	Disables the IPv4 unicast address family for the BGP routing process specified in the previous step. **Note** Routing information for the IPv4 unicast address family is advertised by default for each BGP routing session configured with the **neighbor remote-as** command unless you configure the **no bgp default ipv4-unicast**command before configuring the **neighbor remote-as** command.
Step 5 **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Router(config-router)# neighbor 192.168.99.70 remote-as 65000`	Adds the IP address of the neighbor in the specified autonomous system to the BGP neighbor table of the local router.
Step 6 **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Router(config-router)# neighbor 192.168.99.70 update-source Loopback 0`	Specifies the interface whose IPv4 address is to be used as the source address for the peering. • In the context of this task, the interface must have an IPv4 address with a 32-bit mask configured. Use of a loopback interface is recommended. This address is used to determine the IPv6 next hop by the peer 6PE.
Step 7 **address-family ipv6** [**unicast**] **Example:** `Router(config-router)# address-family ipv6`	Specifies the IPv6 address family and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the router is placed in configuration mode for the IPv6 unicast address family if the **unicast** keyword is not specified with the **address-family ipv6** command.
Step 8 **neighbor** {*ip-address* | *peer-group-name*| *ipv6-address*} **activate** **Example:** `Router(config-router-af)# neighbor 192.168.99.70 activate`	Enables the neighbor to exchange prefixes for the IPv6 address family with the local router.

	Command or Action	Purpose
Step 9	**neighbor** {*ip-address* | *ipv6-address*} **send-label** **Example:** `Router(config-router-af)# neighbor 192.168.99.70 send-label`	Advertises the capability of the router to send MPLS labels with BGP routes. • In IPv6 address family configuration mode this command enables binding and advertisement of aggregate labels when advertising IPv6 prefixes in BGP.

Configuring iBGP Multipath Load Sharing

Perform this task to configure iBGP multipath load sharing and control the maximum number of parallel iBGP routes that can be installed in a routing table.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **maximum-paths ibgp** *number-of-paths*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Router(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.

	Command or Action	Purpose
Step 4	**maximum-paths ibgp** *number-of-paths* **Example:** `Router(config-router)# maximum-paths ibgp 3`	Controls the maximum number of parallel iBGP routes that can be installed in a routing table.

Verifying 6PE Configuration and Operation

When 6PE is running, the following components can be monitored:

- Multiprotocol BGP
- MPLS
- Cisco Express Forwarding for IPv6
- IPv6 routing table

SUMMARY STEPS

1. **show bgp ipv6** {**unicast** | **multicast**} [*ipv6-prefix* / *prefix-length*] [**longer-prefixes**] [**labels**
2. **show bgp ipv6** {**unicast** | **multicast**} **neighbors** [*ipv6-address*] [**received-routes** | **routes** | **flap-statistics** | **advertised-routes** | **paths** *regular-expression* | **dampened-routes**
3. **show mpls forwarding-table** [*network*{*mask*| *length*}| **labels** *label*[**-** *label*] | **interface** *interface*| **nexthop** *address*| **lsp-tunnel**[*tunnel-id*]] [**vrf** *vrf-name*] [**detail**]
4. **show ipv6 cef** [*ipv6-prefix* / *prefix-length*] | [*interface-type interface-number*] [**longer-prefixes** | **similar-prefixes** | **detail** | **internal** | **platform** | **epoch** | **source**]]
5. **show ipv6 route** [*ipv6-address* | *ipv6-prefix* / *prefix-length* | *protocol* | *interface-type interface-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show bgp ipv6** {**unicast**	**multicast**} [*ipv6-prefix* / *prefix-length*] [**longer-prefixes**] [**labels** **Example:** `Router> show bgp ipv6 unicast 2001:DB8:DDDD::/48`	(Optional) Displays entries in the IPv6 BGP routing table. • In this example, information about the IPv6 route for the prefix 2001:DB8:DDDD::/48 is displayed.
Step 2	**show bgp ipv6** {**unicast**	**multicast**} **neighbors** [*ipv6-address*] [**received-routes**	**routes**	**flap-statistics**	**advertised-routes**	**paths** *regular-expression*	**dampened-routes** **Example:** `Router> show bgp ipv6 neighbors unicast 192.168.99.70`	(Optional) Displays information about IPv6 BGP connections to neighbors. • In this example, information including the IPv6 label capability is displayed for the BGP peer at 192.168.99.70.

Command or Action	Purpose
Step 3 **show mpls forwarding-table** [*network*{*mask*| *length*}| **labels** *label*[- *label*] | **interface** *interface*| **nexthop** *address*| **lsp-tunnel**[*tunnel-id*]] [**vrf** *vrf-name*] [**detail**] **Example:** `Router> show mpls forwarding-table`	(Optional) Displays the contents of the MPLS Forwarding Information Base (FIB). • In this example, information linking the MPLS label with IPv6 prefixes is displayed where the labels are shown as aggregate and the prefix is shown as IPv6.
Step 4 **show ipv6 cef** [*ipv6-prefix* / *prefix-length*] | [*interface-type interface-number*] [**longer-prefixes** | **similar-prefixes** | **detail** | **internal** | **platform** | **epoch** | **source**]] **Example:** `Router> show ipv6 cef 2001:DB8:DDDD::/64`	(Optional) Displays FIB entries based on IPv6 address information. • In this example, label information from the Cisco Express Forwarding table for prefix 2001:DB8:DDDD::/64 is displayed.
Step 5 **show ipv6 route** [*ipv6-address* | *ipv6-prefix* / *prefix-length* | *protocol* | *interface-type interface-number* **Example:** `Router> show ipv6 route`	(Optional) Displays the current contents of the IPv6 routing table.

• Output Examples, page 671

Output Examples

Sample Output from the show bgp ipv6 Command

This example shows output information about an IPv6 route using the **show bgp ipv6**command with an IPv6 prefix:

```
Router# show bgp ipv6 2001:DB8:DDDD::/48
BGP routing table entry for 2001:DB8:DDDD::/48, version 15
Paths: (1 available, best #1, table Global-IPv6-Table)
  Not advertised to any peer
  Local
    ::FFFF:192.168.99.70 (metric 20) from 192.168.99.70 (192.168.99.70)
      Origin IGP, localpref 100, valid, internal, best
```

Sample Output from the show bgp ipv6 neighbors Command

This example shows output information about a BGP peer, including the "IPv6 label" capability, using the **show bgp ipv6 neighbors**command with an IP address:

```
Router# show bgp ipv6 neighbors 192.168.99.70
BGP neighbor is 192.168.99.70, remote AS 65000, internal link
  BGP version 4, remote router ID 192.168.99.70
  BGP state = Established, up for 00:05:17
  Last read 00:00:09, hold time is 0, keepalive interval is 60 seconds
```

```
  Neighbor capabilities:
    Route refresh: advertised and received
    Address family IPv6 Unicast: advertised and received
    ipv6 MPLS Label capability: advertised and received
  Received 54 messages, 0 notifications, 0 in queue
  Sent 55 messages, 1 notifications, 0 in queue
  Default minimum time between advertisement runs is 5 seconds
 For address family: IPv6 Unicast
  BGP table version 21880, neighbor version 21880
  Index 1, Offset 0, Mask 0x2
  Route refresh request: received 0, sent 0
  77 accepted prefixes consume 4928 bytes
  Prefix advertised 4303, suppressed 0, withdrawn 1328
  Number of NLRIs in the update sent: max 1, min 0
```

Sample Output from the show mpls forwarding-table Command

This example shows output information linking the MPLS label with prefixes using the **show mpls forwarding-table**command. If the 6PE feature is configured, the labels are aggregated because there are several prefixes for one local label, and the prefix column contains "IPv6" instead of a target prefix.

```
Router# show mpls forwarding-table
Local Outgoing      Prefix            Bytes tag Outgoing       Next Hop
tag   tag or VC     or Tunnel Id      switched  interface
16    Aggregate     IPv6              0
17    Aggregate     IPv6              0
18    Aggregate     IPv6              0
19    Pop tag       192.168.99.64/30  0         Se0/0          point2point
20    Pop tag       192.168.99.70/32  0         Se0/0          point2point
21    Pop tag       192.168.99.200/32 0         Se0/0          point2point
22    Aggregate     IPv6              5424
23    Aggregate     IPv6              3576
24    Aggregate     IPv6              2600
```

Sample Output from the show bgp ipv6 Command

This example shows output information about the top of the stack label with label switching information using the **show bgp ipv6**command with the **labels** keyword:

```
Router# show bgp ipv6 labels
Network              Next Hop              In tag/Out tag
2001:DB8:DDDD::/64  ::FFFF:192.168.99.70   notag/20
```

Sample Output from the show ipv6 cef Command

This example shows output information about labels from the Cisco Express Forwarding table using the **show ipv6 cef**command with an IPv6 prefix:

```
Router# show ipv6 cef 2001:DB8:DDDD::/64
2001:DB8:DDDD::/64
      nexthop ::FFFF:192.168.99.70
      fast tag rewrite with Se0/0, point2point, tags imposed {19 20}
```

Sample Output from the show ipv6 route Command

This example shows output information from the IPv6 routing table using the **show ipv6 route**command. The output shows the IPv6 MPLS virtual interface as the output interface of IPv6 routes forwarded across the MPLS cloud. In this example using the routers in the figure above, the output is from the 6PE1 router.

The 6PE2 router has advertised the IPv6 prefix of 2001:DB8:dddd::/48 configured for the CE2 router and the next-hop address is the IPv4-compatible IPv6 address ::ffff:192.168.99.70, where 192.168.99.70 is the IPv4 address of the 6PE2 router.

```
Router# show ipv6 route
```

```
IPv6 Routing Table - 10 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
B 2001:DB8:DDDD::/64 [200/0]
  via ::FFFF:192.168.99.70, IPv6-mpls
B 2001:DB8:DDDD::/64 [200/0]
  via ::FFFF:192.168.99.70, IPv6-mpls
L 2001:DB8:FFFF::1/128 [0/0]
  via ::, Ethernet0/0
C 2001:DB8:FFFF::/64 [0/0]
  via ::, Ethernet0/0
S 2001:DB8:FFFF::/48 [1/0]
  via 2001:DB8:B00:FFFF::2, Ethernet0/0
```

Configuration Examples for IPv6 over MPLS

The following examples show 6PE configuration examples for three of the routers shown in the figure above and used in the Specifying the Source Address Interface on a 6PE Router, page 665 and Binding and Advertising the 6PE Label to Advertise Prefixes, page 667 sections.

Example: Customer Edge Router

This example shows that the serial interface 0/0 of the customer edge router--CE1 in the figure above--is connected to the service provider and is assigned an IPv6 address. IPv6 is enabled and a default static route is installed using the IPv6 address of serial interface 0/0 of the 6PE1 router.

```
ip cef
!
ipv6 unicast-routing
!
interface Serial 0/0
 description to_6PE1_router
 no ip address
 ipv6 address 2001:DB8:FFFF::2/64
!
ipv6 route ::/0 Serial 0/0 FE80::210:XXXX:FEE1:1001
```

Example: Provider Edge Router

The 6PE router--Router 6PE1 in the figure above--is configured for both IPv4 and IPv6 traffic. Ethernet interface 0/0 is configured with an IPv4 address and is connected to a router in the core of the network--router P1 in the figure above. Integrated IS-IS and TDP configurations on this router are similar to the P1 router.

Router 6PE1 exchanges IPv6 routing information with another 6PE router--Router 6PE2 in the figure above-- using internal BGP (iBGP) established over an IPv4 connection so that all the **neighbor** commands use the IPv4 address of the 6PE2 router. All the BGP peers are within autonomous system 65000, so synchronization with IGP is turned off for IPv4. In IPv6 address family configuration mode, synchronization is disabled by default.

IPv6 and Cisco Express Forwarding for IPv6 are enabled, the 6PE2 neighbor is activated, and aggregate label binding and advertisement is enabled for IPv6 prefixes using the **neighbor send-label** command. Connected and static IPV6 routes are redistributed using BGP. If IPv6 packets are generated in the local router, the IPv6 address for MPLS processing will be the address of loopback interface 0.

This example shows that the serial interface 0/0 connects to the customer and the IPv6 prefix delegated to the customer is 2001:DB8:ffff::/48, which is determined from the service provider IPv6 prefix. A static route is configured to route IPv6 packets between the 6PE route and the CE router.

```
ip cef
ipv6 cef
ipv6 unicast-routing
!
mpls ipv6 source-interface Loopback0
tag-switching tdp router-id Loopback0
!
interface Loopback0
 ip address 192.168.99.5 255.255.255.255
 ipv6 address 2001:DB8:1000:1::1/64
!
interface Ethernet0/0
 description to_P_router
 ip address 192.168.99.1 255.255.255.252
 ip router isis
 tag-switching ip
!
interface Serial0/0
 description to_CE_router
 no ip address
 ipv6 address 2001:DB8:FFFF::1/64
!
router isis
 passive-interface Loopback0
 net 49.0001.1921.6809.9005.00
!
router bgp 65000
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 neighbor 192.168.99.70 remote-as 65000
 neighbor 192.168.99.70 description to_6PE2
 neighbor 192.168.99.70 update-source Loopback0
 !
 address-family ipv6
 neighbor 192.168.99.70 activate
 neighbor 192.168.99.70 send-label
 network 2001:DB8:FFFF::/48
 exit-address-family
!
ipv6 route 2001:DB8:FFFF::/48 Ethernet0/0 2001:DB8:FFFF::2
```

Example: Core Router

This example shows that the router in the core of the network--Router P in the figure above--is running MPLS, IS-IS, and IPv4 only. The Ethernet interfaces are configured with IPv4 address and are connected to the 6PE routers. IS-IS is the IGP for this network and the P1 and 6PE routers are in the same IS-IS area 49.0001. TDP and tag switching are enabled on both the Ethernet interfaces. Cisco Express Forwarding is enabled in global configuration mode.

```
ip cef
!
tag-switching tdp router-id Loopback0
!
interface Loopback0
 ip address 192.168.99.200 255.255.255.255
!
interface Ethernet0/0
 description to_6PE1
 ip address 192.168.99.2 255.255.255.252
 ip router isis
 tag-switching ip
!
interface Ethernet0/1
 description to_6PE2
```

```
 ip address 192.168.99.66 255.255.255.252
 ip router isis
 tag-switching ip
router isis
 passive-interface Loopback0
 net 49.0001.1921.6809.9200.00
```

Where to Go Next

If you want to further customize your MPLS network, refer to the Cisco IOS IP Switching Configuration Guide.

Additional References

Related Documents

Related Topic	Document Title
IPv6 addressing and connectivity	*IPv6 Configuration Guide*
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
IPv6 commands	*Cisco IOS IPv6 Command Reference*
Cisco IOS IPv6 features	Cisco IOS IPv6 Feature Mapping

Standards and RFCs

Standard/RFC	Title
RFCs for IPv6	*IPv6 RFCs*

MIBs

MIB	MIBs Link
CISCO-UNIFIED-FIREWALL-MIB	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing IPv6 over MPLS

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 30 Feature Information for Implementing IPv6 over MPLS

Feature Name	Releases	Feature Information
IPv6 over a Circuit Transport over MPLS	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(15)T 12.3 12.3(2)T 12.4 12.4(2)T	In this feature, communication between the remote IPv6 domains runs native IPv6 protocols over a dedicated link, where the underlying mechanisms are fully transparent to IPv6.
IPv6 Using tunnels Over the Customer Edge Routers	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(15)T 12.3 12.3(2)T 12.4 12.4(2)T	Using tunnels on the CE routers is the simplest way of deploying IPv6 over MPLS networks with no impact on the operation or infrastructure of MPLS.
IPv6 Switching--Provider Edge Router over MPLS (6PE)	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(15)T 12.3 12.3(2)T 12.4 12.4(2)T	The Cisco implementation of IPv6 provider edge router over MPLS enables IPv6 sites to communicate with each other over an MPLS IPv4 core network using MPLS LSPs.

Feature Name	Releases	Feature Information
6PE Multipath	12.2(25)S 12.2(28)SB 12.2(33)SRA 12.4(6)T	The 6PE multipath feature uses multiprotocol internal BGP (MP-iBGP) to distribute IPv6 routes over the MPLS IPv4 core network and to attach an MPLS label to each route.

Implementing IPv6 VPN over MPLS

The Border Gateway Protocol over Multiprotocol Label Switching VPN feature is an implementation of the provider edge (PE)-based VPN model. In principle, there is no difference between IPv4 and IPv6 VPNs. In both IPv4 and IPv6, multiprotocol Border Gateway Protocol (BGP) is the center of the Multiprotocol Label Switching (MPLS) VPN for IPv6 (VPNv6) architecture. It is used to distribute IPv6 routes over the service provider backbone, using the same procedures to work with overlapping addresses, redistribution policies, and scalability issues.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Implementing IPv6 VPN over MPLS

Your network must be running the following Cisco IOS services before you configure IPv6 VPN operation:

- MPLS in provider backbone routers
- MPLS with VPN code in provider routers with VPN PE routers
- BGP in all routers providing a VPN service
- Cisco Express Forwarding switching in every MPLS-enabled router
- Class of Service (CoS) feature

Restrictions for Implementing IPv6 VPN over MPLS

6VPE supports an MPLS IPv4-signaled core. An MPLS IPv6-signaled core is not supported.

Information About Implementing IPv6 VPN over MPLS

IPv6 VPN over MPLS Overview

Multiprotocol BGP is the center of the MPLS IPv6 VPN architecture in both IPv4 and IPv6. It is used to distribute IPv6 routes over the service provider backbone, using the same procedures to work with overlapping addresses, redistribution policies, and scalability issues.

Although IPv6 should not have overlapping address space, IPv6 addresses are prepended with a route distinguisher (RD). A network layer reachability information (NLRI) 3-tuple format (which contains length, IPv6 prefix, and label) is defined to distribute these routes using multiprotocol BGP. The extended community attribute (for example., the route target) is used to control redistribution of routing information by tagging exported routes and filtering imported ones.

For scalability, route reflectors can be used to concentrate routing paths and avoid a full PE mesh. BGP features in IPv6, such as route refresh, automatic route filtering, and outbound route filtering, help reduce the number of routes held in each PE. This document focuses on the following differences between IPv6 and IPv4:

- Creation of a new multiprotocol BGP IPv6 VPN address family and specification of a IPv6 VPN address format
- Specification of a new IPv6 VPN NLRI
- Specification of BGP next-hop encoding when the device has an IPv4-based MPLS core

Some IPv6 VPN features, such as interprovider and Carrier Supporting Carrier (CSC) topologies, are specific to BGP-MPLS IPv6 VPN. Others, such as the link between Autonomous System Boundary Routers (ASBRs), might support IPv4 only, IPv6 only, or both, regardless of the address family being transported.

Addressing Considerations for IPv6 VPN over MPLS

Regardless of the VPN model deployed, an addressing plan must be defined for the VPN that allows hosts to communicate with other sites using one site within one VPN, as well as with public resources.

VPN IPv4 sites often use private addressing for their addressing plan. These addresses do not need to be registered, and they are not routable on the public network. Whenever a host within a private site needs to access a public domain, it goes through a device that finds a public address on its behalf. With IPv4, this can be a network address translator or an application proxy.

Given the larger address space available with IPv6, the easiest approach to IPv6 addressing is to use IPv6 global addresses for the private addressing plan. Another approach is to use unique local addresses (ULAs).

ULAs are easy to filter at site boundaries based on their scope. ULAs are also Internet service provider (ISP)-independent and can be used for communications inside a site without any permanent or intermittent Internet connectivity.

In 6VPE, ULAs are treated as regular global addresses. The device configuration filters ULA prefixes to prevent them from appearing in the public domain. Link-local addresses on the peer will not be announced by BGP (IPv6 or IPv6 VPN) speakers.

A host within a private site that needs to access a public domain can do so through an IPv6 application proxy (such as a web proxy for accessing web pages), which accesses the public resource on the host's behalf with a global routable address, or the host can use a public address of its own. In the latter case, if ULAs have been deployed, the IPv6 host also is configured with a routable global address. A source address selection algorithm is used to select one or the other, based on the destination address.

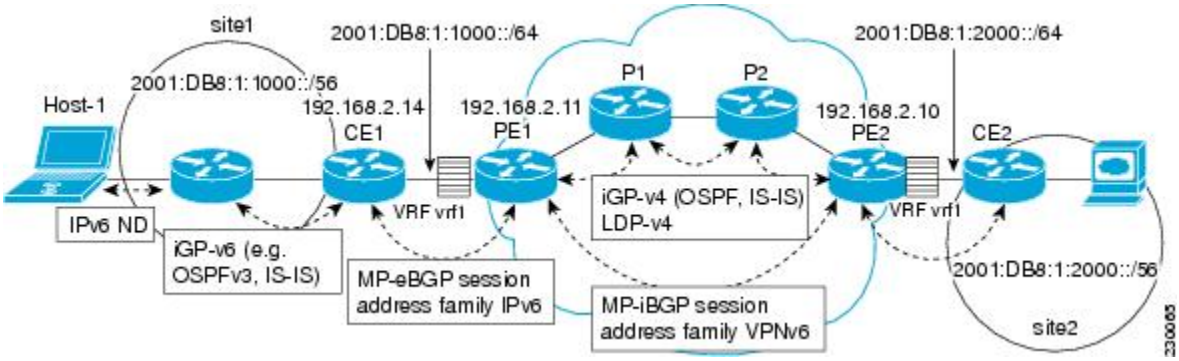
Basic IPv6 VPN over MPLS Functionality

IPv6 VPN takes advantage of the coexistence between IPv6 and IPv4 by leveraging an existent MPLS IPv4 core network:

IPv6 VPN Architecture Overview

The figure below illustrates the important aspects of the IPv6 VPN architecture.

Figure 47 Simple IPv6 VPN Architecture



The CE devices are connected to the provider's backbone using PE devices. The PE devices are connected using provider (P1 and P2 in the figure above) devices. The provider (P) devices are unaware of VPN routes, and, in the case of 6VPE, might support only IPv4. Only PE devices perform VPN-specific tasks. For 6VPE, the PE devices are dual-stack (IPv4 and IPv6) devices.

The routing component of the VPN operation is divided into core routing and edge routing. Core routing, which involves PE devices and P devices, typically is performed by an IPv4 Interior Gateway Protocol (IGP) such as Open Shortest Path First (OSPF) or Intermediate System-to-Intermediate System (IS-IS). In the figure above, the IGP distributes only routes internal to the provider's autonomous system. The core routing enables connectivity among P and PE devices.

Edge routing takes place in two directions: routing between PE pairs and routing between a PE and a CE. Routing between PE pairs is achieved using multiprotocol internal BGP (iBGP) using the IPv6 VPN address family. This method distributes routes learned from CEs through PE-CE routing, using appropriate route export policies at the ingress PE device and appropriate route import policies at the egress PE device.

Routing between the CE and its PE is achieved using a routing protocol that is VPN routing and forwarding (VRF) aware. Static routes, external BGP (eBGP), and Enhanced Interior Gateway Routing Protocol (EIGRP) are VRF-instance aware. In the figure above, eBGP is used between the CE (CE1) and the PE (PE1). At the same time, the CE runs an IPv6 IGP within the VPN site (site1 in the figure above). The CE redistributes IGP routes into multiprotocol-eBGP address family IPv6. At the PE, these routes are installed in the VRF named vrf1, and forwarded to the remote PEs (PE2 in the figure above), according to export policies defined for this VRF.

IPv6 VPN Next Hop

When the device announces a prefix using the MP_REACH_NLRI attribute, the MP-BGP running on one PE inserts a BGP next hop in the update message sent to a remote PE. This next hop is either propagated from the received update (for instance, if the PE is a route reflector), or it is the address of the PE sending the update message (the egress PE).

For the IPv6 VPN address family, the next hop must be an IPv6 VPN address, regardless of the nature of the network between the PE speakers. Because the RD has no significance (the address is not part of any VPN), it is set to 0. If the provider network is a native IPv6 network, the remaining part of the next hop is the IPv6 address of the egress PE. Otherwise, it is an IPv4 address used as an IPv6-mapped address (for example, ::FFFF:IPv4-address).

MPLS Forwarding

When it receives IPv6 traffic from one customer site, the ingress PE device uses MPLS to tunnel IPv6 VPN packets over the backbone toward the egress PE device identified as the BGP next hop. The ingress PE device prepends the IPv6 packets with the outer and inner labels before putting the packet on the egress interface.

Under normal operation, a P device along the forwarding path does not look inside the frame beyond the first label. The P device either swaps the incoming label with an outgoing one or removes the incoming label if the next device is a PE device. Removing the incoming label is called penultimate hop popping. The remaining label (BGP label) is used to identify the egress PE interface toward the customer site. The label also hides the protocol version (IPv6) from the last P device, which it would otherwise need to forward an IPv6 packet.

A P device is ignorant of the IPv6 VPN routes. The IPv6 header remains hidden under one or more MPLS labels. When the P device receives an MPLS-encapsulated IPv6 packet that cannot be delivered, it has two options. If the P device is IPv6 aware, it exposes the IPv6 header, builds an Internet Control Message Protocol (ICMP) for IPv6 message, and sends the message, which is MPLS encapsulated, to the source of the original packet. If the P device is not IPv6 aware, it drops the packet.

- 6VPE over GRE Tunnels, page 682

6VPE over GRE Tunnels

In some Cisco software releases, the ingress PE device uses IPv4 generic routing encapsulation (GRE) tunnels combined with 6VPE over MPLS to tunnel IPv6 VPN packets over the backbone toward the egress PE device identified as the BGP next hop.

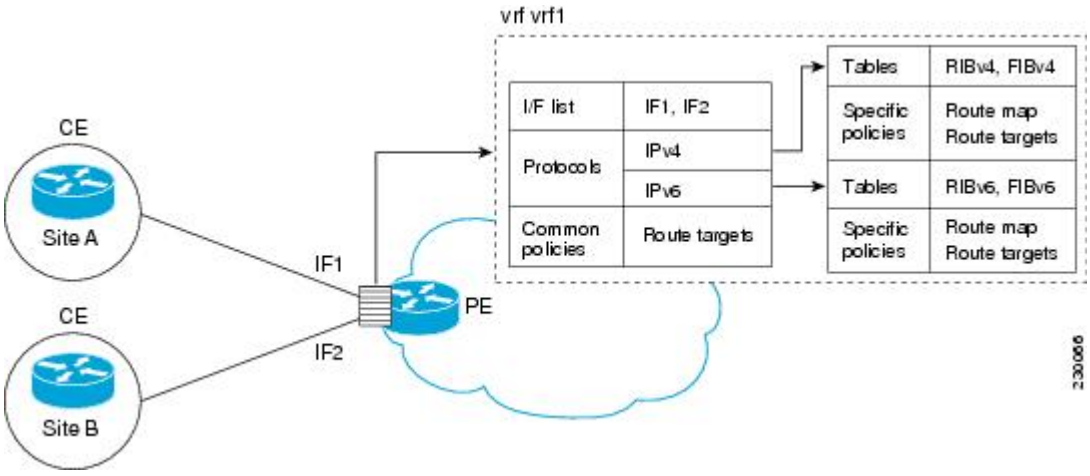
VRF Concepts

A virtual routing and forwarding (VRF) entity works with a private customer-specific Routing Information Base (RIB) and Forwarding Information Base (FIB). Although IPv4 and IPv6 routing tables are distinct, it is convenient for the two protocols to share the same VRF for a specific customer.

IPv6 VPN customers are likely to be existing VPNv4 customers that are either deploying dual-stack hosts and devices or shadowing some of their IPv4 infrastructure with IPv6 nodes. Several deployment models are possible. Some customers use separate logical interfaces for IPv4 and IPv6 and define separate VRFs on each. Although this approach provides flexibility to configure separate policies for IPv4 and IPv6, it prevents sharing the same policy. Another approach, the multiprotocol VRF, keeps a single VRF on the PE-CE interface, and enables it for IPv4, IPv6, or both. It is then possible to define common or separate policies for each IP version. With this approach, a VRF is better defined as the set of tables, interfaces, and policies found at the PE, and is used by sites of a particular VPN connected to this PE.

The figure below illustrates the multiprotocol VRF, in which the VRF named vrf1is enabled for both IPv4 and IPv6 and is associated with two interfaces (IF1, IF2), two sets of tables (IPv4 RIB and FIB and IPv6 RIB and FIB), and a set of common or distinct policies.

Figure 48 Multiprotocol VRF



IPv6 VPN Scalability

PE-based VPNs such as BGP-MPLS IPv6 VPN scale better than CE-based VPNs. A network designer must consider scaling when designing the network. The following points need to be considered:

- Routing table size, which includes the size of VRF tables and BGP tables
- Number of BGP sessions, which grows as a square number of PEs

Routing table size concerns occur with PEs that handle many customer sites. Not only do these PEs have one RIB and FIB per connected customer, but also the PEs' BGP tables, which total all entries from individual VRFs, grow accordingly. Another scalability problem occurs when the number of PEs in the provider network grows beyond a certain level. Assuming that a significant number of sites belonging to the same VPN are spread over many PEs, the number of multiprotocol BGP sessions may rapidly become prohibitive: (n -1) x n /2, where n is the number of PEs.

The following features are included in IPv6 VPN over MPLS:

- Route refresh and automatic route filtering--Limits the size of routing tables, because only routes imported into a VRF are kept locally. When the import policy changes, a route refresh can be sent to query a retransmission of routing updates.
- Outbound route filtering (ORF)--Allows the ingress PE to advertise filters to the egress PE so that updates are not sent unnecessarily over the network.
- Route reflectors--Route reflectors (RRs) are iBGP peers that propagate iBGP routes learned from other iBGP peers. RRs are used to concentrate iBGP sessions.

Advanced IPv6 MPLS VPN Functionality

Advanced MPLS features such as accessing the Internet from a VPN for IPv4, multiautonomous-system backbones, and CSCs are generally the same for IPv6 as for IPv4. However, there are differences in addressing and in the way 6VPE operates over an IPv4 backbone.

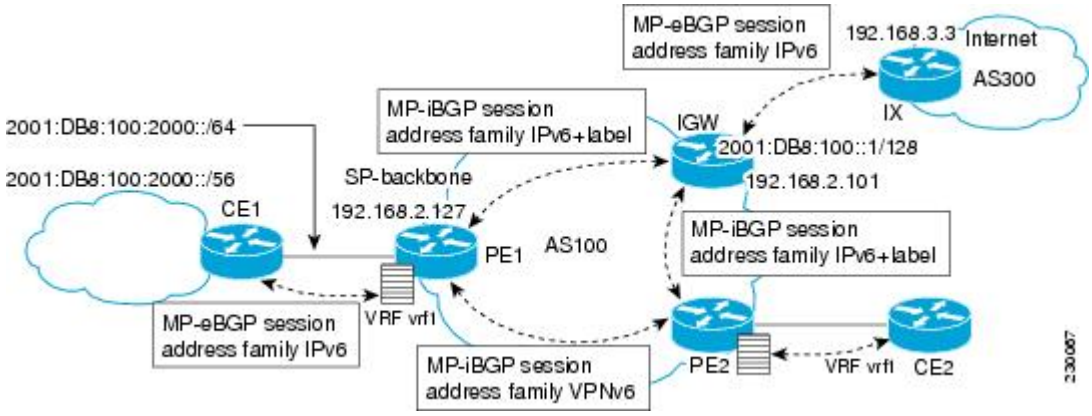
The following sections describe concepts for advanced IPv6 MPLS VPN functionality:

Internet Access

Most VPN sites require access to the Internet. RFC 4364 describes a set of models for enabling IPv4 and IPv6 VPN access to the Internet. In one model, one interface is used by the CE to connect to the Internet and a different one to connect to the VRF. Another model is in which all Internet routes are redistributed into the VRF; however, this approach has the disadvantage of requiring the Internet routes be replicated in each VRF.

In one scenario, a static route is inserted into the VRF table, with a next hop that points to the Internet gateway found in the IPv6 default table. The figure below illustrates this scenario, in which Internet access is provided to the customer in the VRF named vrf1.

Figure 49 Internet Access Topology



A customer site that has access public resources over the Internet must be known by a public prefix. Unlike IPv4, IPv6 does not offer a Network Address Translation (NAT) mechanism that translates private addresses into public addresses when leaving the site boundaries. This implies that hosts within the site speak with public addresses and appear in the public domain.

For outbound traffic, the default route configured in the VRF table at ingress PE (PE1) directs traffic for destinations outside the VPN to the Internet gateway.

For inbound traffic, a route must exist at the Internet gateway to direct the traffic for a customer site via its PE of attachment (PE1 in the figure above). This route can be distributed by the ingress PE (PE1) using multiprotocol iBGP (with the IPv6 address family configuration), so no specific configuration is needed on a per-VPN PE basis at the Internet gateway. Nevertheless, for inbound traffic at PE1, a route must exist in the default table for the customer site global prefix pointing to the VRF of the site.

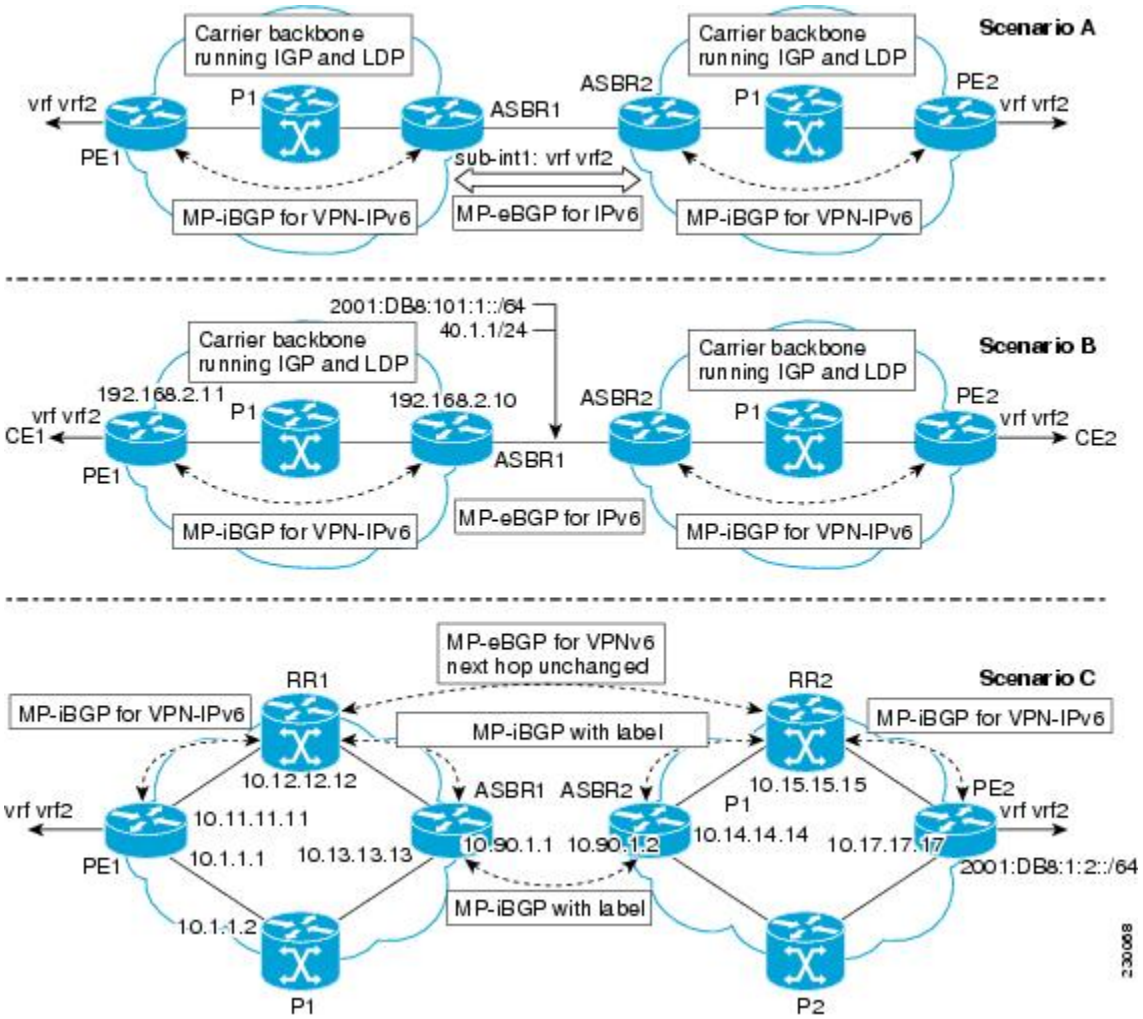
Multiautonomous-System Backbones

The problem of interprovider VPNs is similar for IPv6 and IPv4, assuming that IPv6 was deployed everywhere IPv4 was deployed.

In IPv6 deployments that cross autonomous system boundaries, providers may have to obtain a peering model, or work with the peering model put in place for VPNv4.

The figure below illustrates interprovider scenarios in IPv6 VPN.

Figure 50 Interprovider Scenarios



Depending on the network protocol used between ASBRs, the three scenarios shown in the figure above can have several implementation options. For instance, scenario B, which suggests a multiprotocol eBGP IPv6 VPN peering between ASBRs, could use either an IPv6 or an IPv4 link.

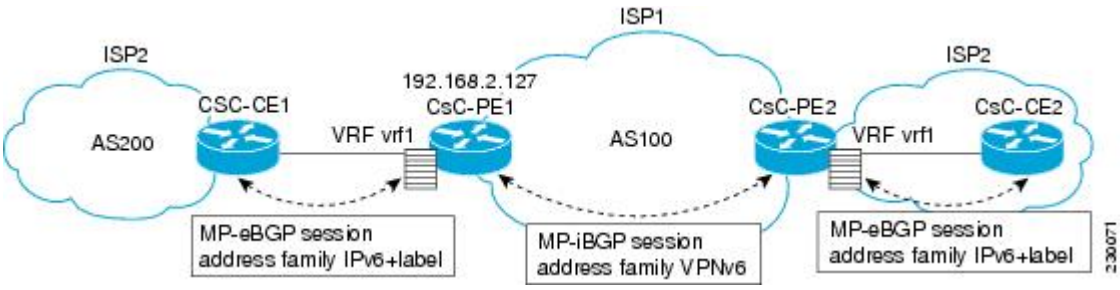
In scenario C, multihop multiprotocol eBGP redistributes IPv6 VPN routes across route reflectors in different autonomous systems. Labeled IPv4 routes to the PEs (in the 6VPE case) need to be advertised across ASBRs so that a complete labeled switch path is set up end to end.

Carrier Supporting Carriers

The CSC feature provides VPN access to a customer service provider, so this service needs to exchange routes and send traffic over the ISP MPLS backbone. The only difference from a regular PE is that it provides MPLS-to-MPLS forwarding on the CSC-CE to CSC-PE interface, rather than IP-to-MPLS forwarding.

The figure below highlights the two ISPs' interface.

Figure 51 CSC 6VPE Configuration Example



BGP IPv6 PIC Edge for IP MPLS

The BGP IPv6 PIC Edge for IP MPLS feature improves convergence for both core and edge failures after a network failure. The BGP IPv6 prefix-independent convergence (PIC) edge for IP MPLS feature creates and stores a backup or alternate path in the RIB, FIB, and in Cisco Express Forwarding, so that the backup or alternate path can immediately take over wherever a failure is detected, thus enabling fast failover.

How to Implement IPv6 VPN over MPLS

Configuring a Virtual Routing and Forwarding Instance for IPv6

A VRF is an address family-independent object that can be enabled and configured for each of the supported address families. Configuring a VRF consists of the following three steps:

- Configuring the address-family-independent part of the VRF
- Enabling and configuring IPv4 for the VRF
- Enabling and configuring IPv6 for the VRF

A VRF is given a name and an RD. The RD is configured outside the context of the address family, although the RD is used to distinguish overlapping addresses within the context of a particular BGP address family. Having separate RDs for IPv4 VPN addresses and IPv6 VPN addresses does not matter. On Cisco devices, the RDs are the same to simplify configuration and VPN management.

Users can configure policies in common between IPv4 and IPv6 when not using an address family context. This feature is shared route targets (import and export), and it is useful in a migration scenario, where IPv4 policies already are configured and IPv6 policies should be the same as the IPv4 policies.

The IPv4 and IPv6 address family can each be enabled and configured separately. Note that the route-target policies entered at this level override global policies that might have been specified during address family-independent configuration.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mls ipv6 vrf**
4. **vrf definition** *vrf-name*
5. **rd** *route-distinguisher*
6. **route-target** {**import**| **export**| **both**} *route-target-ext-community*
7. **exit**
8. **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*]
9. **route-target** {**import** | **export** | **both**} *route-target-ext-community*
10. **exit**
11. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast**]
12. **route-target** {**import**| **export**| **both**} *route-target-ext-community*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mls ipv6 vrf** **Example:** `Device(config)# mls ipv6 vrf`	Enables IPv6 globally in a VRF.
Step 4	**vrf definition** *vrf-name* **Example:** `Device(config)# vrf definition vrf1`	Configures a VPN VRF routing table and enters VRF configuration mode.
Step 5	**rd** *route-distinguisher* **Example:** `Device(config-vrf)# rd 100:1`	Specifies the RD for a VRF.
Step 6	**route-target** {**import**| **export**| **both**} *route-target-ext-community* **Example:** `Device(config-vrf)# route target import 100:10`	Specifies the route target VPN extended communities for both IPv4 and IPv6.
Step 7	**exit** **Example:** `Device(config-vrf)# exit`	Exits VRF configuration mode.
Step 8	**address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] **Example:** `Device(config)# address-family ipv4`	Enters address family configuration mode to configure a routing session using standard IPv4 address prefixes.

	Command or Action	Purpose
Step 9	**route-target** {**import** | **export** | **both**} *route-target-ext-community* **Example:** `Device(config-vrf-af)# route target import 100:11`	Specifies the route target VPN extended communities specific to IPv4.
Step 10	**exit** **Example:** `Device(config-vrf-af)# exit`	Exits address family configuration mode on this VRF.
Step 11	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast**] **Example:** `Device(config-vrf)# address-family ipv6`	Enters address family configuration mode for configuring routing sessions such as BGP that use standard IPv6 address prefixes.
Step 12	**route-target** {**import**| **export**| **both**} *route-target-ext-community* **Example:** `Device(config-vrf-af)# route target import 100:12`	Specifies the route target VPN extended communities specific to IPv6.

Binding a VRF to an Interface

To specify which interface belongs to which VRF, use the **vrf forwarding** command for both IPv4 and IPv6. An interface cannot belong to more than one VRF. When the interface is bound to a VRF, previously configured addresses (IPv4 and IPv6) are removed, and they must be reconfigured.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **vrf forwarding** *vrf-name*
5. **ip address** *ip-address mask* [**secondary**]
6. **ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits* / *prefix-length*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface Ethernet 0/0`	Specifies an interface type and number, and places the device in interface configuration mode.
Step 4	**vrf forwarding** *vrf-name* **Example:** `Device(config-if)# vrf forwarding vrf1`	Associates a VPN VRF with an interface or subinterface. • Any address, IPv4 or IPv6, that was configured prior to entering this command will be removed.
Step 5	**ip address** *ip-address mask* [**secondary**] **Example:** `Device(config-if)# ip address 10.10.10.1 255.255.255.0`	Configures an IPv4 address on the interface.
Step 6	**ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits* / *prefix-length*} **Example:** `Device(config-if)# ipv6 address 2001:DB8:100:1::1/64`	Configures an IPv6 address on the interface.

Configuring a Static Route for PE-to-CE Routing

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 route** [**vrf** *vrf-name*] *ipv6-prefix* / *prefix-length* {*ipv6-address* | *interface-type interface-number* [*ipv6-address*]} [**nexthop-vrf** [*vrf-name1* | **default**]] [*administrative-distance*] [*administrative-multicast-distance* | **unicast** | **multicast**] [*next-hop-address*] [**tag** *tag*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 route** [**vrf** *vrf-name*] *ipv6-prefix* / *prefix-length* {*ipv6-address* | *interface-type interface-number* [*ipv6-address*]} [**nexthop-vrf** [*vrf-name1* | **default**]] [*administrative-distance*] [*administrative-multicast-distance* | **unicast** | **multicast**] [*next-hop-address*] [**tag** *tag*] **Example:** `Device(config)# ipv6 route vrf vrf1 ::/0 2001:DB8:200::1 nexthop-vrf default`	Installs the specified IPv6 static route using the specified next hop.

Configuring eBGP PE-to-CE Routing Sessions

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast**]
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
6. **neighbor** { *p-address* | *peer-group-name* | ipv6-address} **activate**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast**] **Example:** `Device(config-router)# address-family ipv6 vrf vrf1`	Enters address family configuration mode.
Step 5	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router-af)# neighbor 2001:DB8:100:1::2 remote-as 200`	Adds an entry to the multiprotocol BGP neighbor table.
Step 6	**neighbor** { *p-address* | *peer-group-name* | ipv6-address} **activate** **Example:** `Device(config-router-af)# neighbor 2001:DB8:100:1::2 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.

Configuring the IPv6 VPN Address Family for iBGP

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **address-family vpnv6** [**unicast**]
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
8. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**]
9. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.11 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table. • In IPv6 VPN, the peer address typically is an IPv4 address, in order to enable the BGP session to be transported over the IPv4-based core network.

	Command or Action	Purpose
Step 5	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.11 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 6	**address-family vpnv6** [**unicast**] **Example:** `Device(config-router)# address-family vpnv6`	Places the device in address family configuration mode for configuring routing sessions.
Step 7	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.11 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 8	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**] **Example:** `Device(config-router-af)# neighbor 192.168.2.11 send-community extended`	Specifies that a communities attribute should be sent to the BGP neighbor.
Step 9	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode.

Configuring Route Reflectors for Improved Scalability

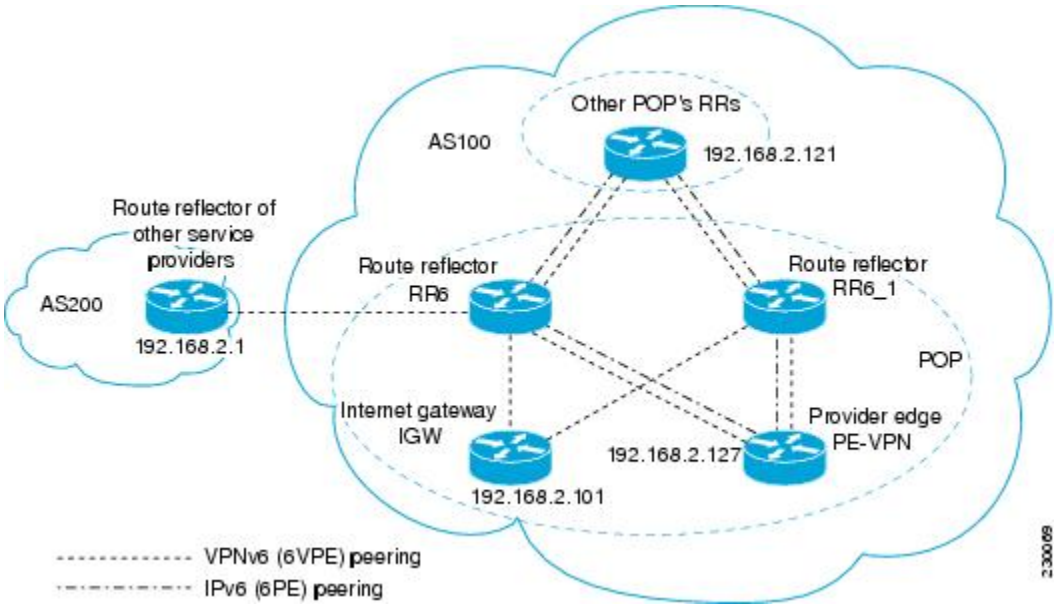
In this task, two RRs are configured for redundancy reasons. Deploying RRs improves scalability by drastically reducing the number of BGP sessions. One RR usually peers with many iBGP speakers, preventing a full mesh of BGP sessions.

In an MPLS-based core, RRs are not part of the label switch paths and can be located anywhere in the network. For example, in a flat RR design, RRs can be deployed at Level 1 points of presence (POPs) and peer together in a full-mesh topology. In a hierarchical RR design, RRs could be deployed at Level 1 and Level 2 POPs, with Level 1 POPs peering together and with Level 2 RRs.

In a typical case where 6VPE is deployed in a preexisting MPLS network (for example, providing VPNv4 services), it is likely that some RR design is already in place, and a similar RR infrastructure for IPv6 VPN

services can be deployed. The figure below illustrates the main peering points between the RR in the ISP POP and the set of its RR clients.

Figure 52 Route Reflector Peering Design



The following list of BGP RR clients must be configured at each IPv6 RR (RR6 and RR6_1 in the figure above) device, at each POP:

- PE devices (PE-VPN) of the POP providing IPv6 VPN access to the ISP customers. This includes both IPv6 VPN (6VPE) peering for interconnecting customer sites and IPv6 peering (6PE) for providing Internet access to VPN customers (see the Configuring Internet Access, page 702).
- Internet gateway (IGW) located in the POP in order to provide PE customers with access to the IPv6 Internet (see the Configuring Internet Access, page 702).
- RRs from other service providers. This feature is used to provide interautonomous-system connectivity, and it includes both IPv6 and IPv6 VPN peering. This service is described in the Configuring a Multiautonomous-System Backbone for IPv6 VPN, page 710 section.
- RRs in other POPs. All RRs peer together, with both IPv6 and IPv6 VPN address families enabled.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
7. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
8. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
9. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
10. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
11. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
12. **neighbor** {*ip-address* | *ipv6-address*| *peer-group-name*} **ebgp-multihop** [*ttl*]
13. **address-family ipv6**
14. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
15. **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label**
16. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client**
17. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
18. **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label**
19. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client**
20. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
21. **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label**
22. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client**
23. **exit**
24. **address-family vpnv6** [**unicast**]
25. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
26. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**]
27. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client**
28. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
29. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**]
30. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client**
31. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
32. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**]
33. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client**
34. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **next-hop-unchanged** [**allpaths**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.101 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table, and provides peering with the Internet gateway in order to provide Internet access.
Step 5	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.101 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 6	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.121 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table, and provides peering with the other POP's RR.

	Command or Action	Purpose
Step 7	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.121 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 8	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.127 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table.
Step 9	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.127 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 10	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.1 remote-as 200`	(Optional) Adds an entry to the multiprotocol BGP neighbor table, and provides peering with the RR of the peer ISP in order to provide inter-VPN service.
Step 11	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.1 update-source Loopback 0`	(Optional) Enables the BGP session to use a source address on the specified interface.
Step 12	**neighbor** {*ip-address* | *ipv6-address*| *peer-group-name*} **ebgp-multihop** [*ttl*] **Example:** `Device(config-router)# neighbor 192.168.2.1 ebgp-multihop`	(Optional) Accepts and attempts BGP connections to external peers residing on networks that are not directly connected.

	Command or Action	Purpose
Step 13	**address-family ipv6** **Example:** `Device(config-router)# address-family ipv6`	(Optional) Enters address family configuration mode in order to provide Internet access service.
Step 14	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.101 activate`	(Optional) Enables the exchange of information for this address family with the specified neighbor.
Step 15	**neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label** **Example:** `Device(config-router-af)# neighbor 192.168.2.101 send-label`	(Optional) Enables a BGP device to send MPLS labels with BGP routes to a neighboring BGP device.
Step 16	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client** **Example:** `Device(config-router-af)# neighbor 192.168.2.101 route-reflector-client`	(Optional) Configures the device as a BGP route reflector and configures the specified neighbor as its client.
Step 17	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.121 activate`	(Optional) Enables the exchange of information for this address family with the specified BGP neighbor.
Step 18	**neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label** **Example:** `Device(config-router-af)# neighbor 192.168.2.121 send-label`	(Optional) Enables a BGP device to send MPLS labels with BGP routes to a neighboring BGP device.
Step 19	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client** **Example:** `Device(config-router-af)# neighbor 192.168.2.121 route-reflector-client`	(Optional) Configures the specified neighbor as a route reflector client.

Command or Action	Purpose
Step 20 **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.127 activate`	(Optional) Enables the exchange of information for this address family with the specified BGP neighbor.
Step 21 **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label** **Example:** `Device(config-router-af)# neighbor 192.168.2.127 send-label`	(Optional) Enables a BGP device to send MPLS labels with BGP routes to a neighboring BGP device.
Step 22 **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client** **Example:** `Device(config-router-af)# neighbor 192.168.2.127 route-reflector-client`	(Optional) Configures the specified neighbor as a route reflector client.
Step 23 **exit** **Example:** `Device(config-router-af)# exit`	(Optional) Exits address family configuration mode.
Step 24 **address-family vpnv6** [**unicast**] **Example:** `Device(config-router)# address-family vpnv6`	Places the device in address family configuration mode for configuring routing sessions.
Step 25 **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.121 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 26 **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**] **Example:** `Device(config-router-af)# neighbor 192.168.2.21 send-community extended`	Specifies that a communities attribute should be sent to the BGP neighbor.

	Command or Action	Purpose
Step 27	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client** **Example:** `Device(config-router-af)# neighbor 192.168.2.121 route-reflector-client`	Configures the specified neighbor as a route reflector client.
Step 28	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.127 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 29	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**] **Example:** `Device(config-router-af)# neighbor 192.168.2.127 send-community extended`	Specifies that a communities attribute should be sent to the BGP neighbor.
Step 30	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client** **Example:** `Device(config-router-af)# neighbor 192.168.2.127 route-reflector-client`	Configures the specified neighbor as a route reflector client.
Step 31	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.1 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 32	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**] **Example:** `Device(config-router-af)# neighbor 192.168.2.1 send-community extended`	Specifies that a communities attribute should be sent to the BGP neighbor.

	Command or Action	Purpose
Step 33	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client** **Example:** `Device(config-router-af)# neighbor 192.168.2.1 route-reflector-client`	Configures the specified neighbor as a route reflector client.
Step 34	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **next-hop-unchanged** [**allpaths**] **Example:** `Device(config-router-af)# neighbor 192.168.2.1 next-hop-unchanged allpaths`	Enables an EBGP multihop peer to propagate to the next hop unchanged for paths.

Configuring Internet Access

Customers with IPv6 VPN access need to have access to the Internet through IPv6. The design of this service is similar to a global Internet access service. 6VPE devices located in a Level 1 POP (colocated with an IGW device) can access the IGW natively, whereas 6VPE devices located in Level 2 and Level 3 POPs with no direct access to the IGW can access the IGW in their closest Level 1 POP over 6PE.

Configuring VPN Internet access in such a 6VPE device involves configuring BGP peering with the IGW (in most cases through the IPv6 RR, as described in the Configuring Route Reflectors for Improved Scalability section). Then the user must configure cross-table routing to enable communication between the private domain (the VRF) and the public domain (the Internet).

The figure above illustrates the following configuration tasks:

- Configuring the Internet Gateway, page 702
- Configuring the IPv6 VPN PE, page 707

Configuring the Internet Gateway

- Configuring iBGP 6PE Peering to the VPN PE, page 702
- Configuring the Internet Gateway as the Gateway to the Public Domain, page 704
- Configuring eBGP Peering to the Internet, page 705

Configuring iBGP 6PE Peering to the VPN PE

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **address-family ipv6**
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
8. **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.127 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table to provide peering with the VPN PE.

	Command or Action	Purpose
Step 5	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.127 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 6	**address-family ipv6** **Example:** `Device(config-router)# address-family ipv6`	Enters address family configuration mode in order to exchange global table reachability.
Step 7	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.127 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 8	**neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label** **Example:** `Device(config-router-af)# neighbor 192.168.2.127 send-label`	Enables a BGP device to send MPLS labels with BGP routes to a neighboring BGP device, and allows the PE VPN to reach the Internet gateway over MPLS.

Configuring the Internet Gateway as the Gateway to the Public Domain

Use the 6PE peering configuration established in the Configuring iBGP 6PE Peering to the VPN PE, page 702 to perform this task.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv6**
5. **network** *ipv6-address* / *prefix-length*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**address-family ipv6** **Example:** `Device(config-router)# address-family ipv6`	Enters address family configuration mode in order to exchange global table reachability.
Step 5	**network** *ipv6-address* / *prefix-length* **Example:** `Device(config-router-af)# network 2001:DB8:100::1/128`	Configures the network source of the next hop to be used by the PE VPN.

Configuring eBGP Peering to the Internet

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **address-family ipv6**
6. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
7. **aggregate-address** *address mask* [**as-set**] [**summary-only**] [**suppress-map** *map-name*] [**advertise-map** *map-name*] [**attribute-map** *map-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor FE80::300::1 Ethernet0/0 remote-as 300`	Adds an entry to the multiprotocol BGP neighbor table, and provides peering with PE (PE-VPN). • The peering is done over link-local addresses.
Step 5	**address-family ipv6** **Example:** `Device(config-router)# address-family ipv6`	Enters address family configuration mode in order to exchange global table reachability.
Step 6	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor FE80::300::1 Ethernet0/0 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.

	Command or Action	Purpose
Step 7	**aggregate-address** *address mask* [**as-set**] [**summary-only**] [**suppress-map** *map-name*] [**advertise-map** *map-name*] [**attribute-map** *map-name*] **Example:** `Device(config-router-af)# aggregate-address 2001:DB8::/32 summary-only`	Creates an aggregate prefix before advertising it to the Internet.

Configuring the IPv6 VPN PE

Configuring a Default Static Route from the VRF to the Internet Gateway

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 route** [**vrf** *vrf-name*] *ipv6-prefix/prefix-length* {*ipv6-address* | *interface-type interface-number* [*ipv6-address*]} [**nexthop-vrf** [*vrf-name1* | **default**]] [*administrative-distance*] [*administrative-multicast-distance* | **unicast** | **multicast**] [*next-hop-address*] [**tag** *tag*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Command or Action	Purpose
Step 3 **ipv6 route** [**vrf** *vrf-name*] *ipv6-prefix*/*prefix-length* {*ipv6-address* | *interface-type interface-number* [*ipv6-address*]} [**nexthop-vrf** [*vrf-name1* | **default**]] [*administrative-distance*] [*administrative-multicast-distance* | **unicast** | **multicast**] [*next-hop-address*] [**tag** *tag*] **Example:** `Device(config)# ipv6 route vrf vrf1 ::/0 2001:DB8:100::1 nexthop-vrf default`	Configures a default static route from the VRF to the Internet gateway to allow outbound traffic to leave the VRF.

Configuring a Static Route from the Default Table to the VRF

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 route** [**vrf** *vrf-name*] *ipv6-prefix*/*prefix-length* {*ipv6-address* | *interface-type interface-number* [*ipv6-address*]} [**nexthop-vrf** [*vrf-name1* | **default**]] [*administrative-distance*] [*administrative-multicast-distance* | **unicast** | **multicast**] [*next-hop-address*] [**tag** *tag*]

DETAILED STEPS

Command or Action	Purpose
Step 1 **enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2 **configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3 **ipv6 route** [**vrf** *vrf-name*] *ipv6-prefix*/*prefix-length* {*ipv6-address* | *interface-type interface-number* [*ipv6-address*]} [**nexthop-vrf** [*vrf-name1* | **default**]] [*administrative-distance*] [*administrative-multicast-distance* | **unicast** | **multicast**] [*next-hop-address*] [**tag** *tag*] **Example:** `Device(config)# ipv6 route 2001:DB8:100:2000::/64 nexthop-vrf vrf1`	Configures a static route from the default table to the VRF to allow inbound traffic to reach the VRF.

Configuring iBGP 6PE Peering to the Internet Gateway

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast**]
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
8. **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label**
9. **network** *ipv6-address / prefix-length*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.101 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table for peering with the Internet gateway.

Command or Action	Purpose
Step 5 **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.101 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 6 **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast**] **Example:** `Device(config-router)# address-family ipv6`	Enters address family configuration mode to exchange global table reachability.
Step 7 **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.101 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 8 **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label** **Example:** `Device(config-router-af)# neighbor 192.168.2.101 send-label`	Enables label exchange for this address family to this neighbor to enable the VPN PE to reach the Internet gateway over MPLS.
Step 9 **network** *ipv6-address* / *prefix-length* **Example:** `Device(config-router-af)# network 2001:DB8:100:2000::/64`	Provides the VRF prefix to the Internet gateway.

Configuring a Multiautonomous-System Backbone for IPv6 VPN

Two VPN sites may be connected to different autonomous systems because the sites are connected to different service providers. The PE devices attached to that VPN is then unable to maintain iBGP connections with each other or with a common route reflector. In this situation, there must be some way to use eBGP to distribute VPN-IPv6 addresses.

The following configuration example illustrates two scenarios, one in which a multiprotocol eBGP-IPv6 VPN peering between ASBRs uses an IPv4 link, and the same scenario using an IPv6 link. If the peering between ASBRs is performed over an IPv4 link, the BGP configuration on ASBR1 is as follows:

```
router bgp 1001
no bgp default ipv4-unicast
no bgp default route-target filter
neighbor 192.1.1.1 remote-as 1002
neighbor 192.168.2.11 remote-as 1001
neighbor 192.168.2.11 update-source Loopback1
```

```
!
address-family vpnv6
!Peering to ASBR2 over an IPv4 link
neighbor 192.1.1.1 activate
neighbor 192.1.1.1 send-community extended
!Peering to PE1 over an IPv4 link
neighbor 192.168.2.11 activate
neighbor 192.168.2.11 next-hop-self
 neighbor 192.168.2.11 send-community extended
```

If the peering between ASBRs is performed over an IPv6 link, the BGP configuration on ASBR1 is as follows:

```
router bgp 1001
neighbor 2001:DB8:101::72d remote-as 1002
!
 address-family vpnv6
!Peering to ASBR2 over an IPv6 link
 neighbor 2001:DB8:101::72d activate
 neighbor 2001:DB8:101::72d send-community extended
```

The next several tasks describe how to configure the PE VPN for a multiautonomous-system backbone using multihop multiprotocol eBGP to redistribute VPN routes across RRs in different autonomous systems. Labeled IPv4 routes to the PEs are advertised across ASBRs so that a complete label switch path (LSP) is set up end to end.

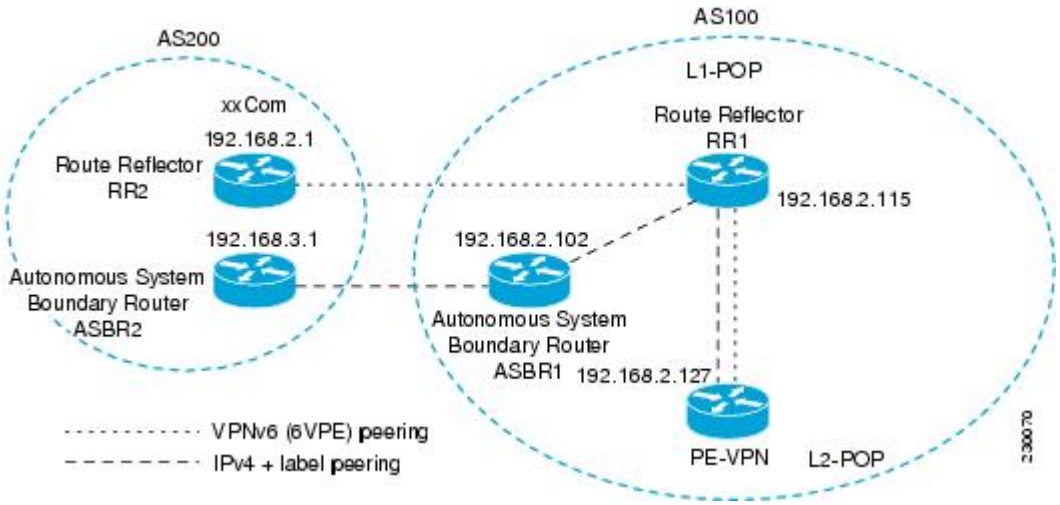
In this scenario, the ASBRs are not VPN aware; only the RRs are VPN aware. The following configuration should be available and understood:

- The ASBRs are providing the PEs' loopback addresses to service providers they peer with. That includes:
  - The VPN PE's IPv4 loopback address (/32) for enabling next-hop resolution at the remote service provider location.
  - The VPN RR's IPv4 loopback address (/32) for enabling interprovider (inter-RR) eBGP peering.
- For the VPN PE's IPv4 loopback address, the address providing is performed over multiprotocol BGP, with the label, up to the remote PEs, so that the label establishes an end-to-end LSP. Therefore, the following MP-BGP peering was set up for VPNv4:
  - VPN PEs are iBGP peering with VPN RRs.
  - ASBRs are iBGP peering with VPN RRs.
  - ASBRs are eBGP peering with the remote service provider ASBR.
- The VPN RRs of each service provider are peering together over eBGP and exchanging VPN routes. The next hop is forwarded unchanged, so that the end-to-end LSP is not via RRs.

To enable IPv6 VPN interautonomous-system access in this scenario, the ISP needs to modify the configurations at the PE VPN and at the RR. The same RRs are set up to provide a similar service for VPNv4. In that context, because the peering between the RR and the ASBR and between ASBRs is solely to exchange labels for IPv4 next hops used by both IPv4 VPN and IPv6 VPN, the ASBRs remain completely IPv6 unaware, and no configuration change is required there.

The figure below shows the BGP peering points required to enable IPv6 interprovider connectivity from the PE-VPN device (providing IPv6 VPN access) to the xxCom network.

Figure 53 BGP Peering Points for Enabling Interautonomous System Scenario C



The following additional BGP peerings are necessary to enable interautonomous-system communication from the IPv6 VPN PE located in the Level 2 POP:

- IPv4 with label peering from the PE VPN to the route reflector named RR1 (which is already configured if VPNv4 interautonomous system is deployed on the same nodes, using the same LSP).
- IPv4 with label peering from RR1 to ASBR1.
- IPv4 with label peering between ASBR1 and ASBR2.
- IPv6 VPN peering between RR1 and RR2 (which is the route reflector in the other autonomous systems) to exchange IPv6 VPN routes.
- IPv6 VPN peering with RR1. If the same route reflectors used to scale the IPv6 VPN service are used for interautonomous-system capability, then this function might also be already configured (see the Configuring Route Reflectors for Improved Scalability, page 694).

Configuring the multiautonomous-system backbone for IPv6 VPN consists of the following tasks:

Configuring the PE VPN for a Multiautonomous-System Backbone

Configuring iBGP IPv6 VPN Peering to a Route Reflector

Perform this task to configure iBGP IPv6 VPN peering to a route reflector named RR1.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **address-family vpnv6** [**unicast**]
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
8. **neighbor** {*ip-address* | *ipv6-address* | **peer-group-name**} **send-community** [**both** | **standard** | **extended**]
9. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.115 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table for peering with the route reflector with interautonomous-system functionality.

Command or Action	Purpose
Step 5 **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.115 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 6 **address-family vpnv6** [**unicast**] **Example:** `Device(config-router)# address-family vpnv6`	(Optional) Places the device in address family configuration mode for configuring routing sessions.
Step 7 **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.115 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 8 **neighbor** {*ip-address* | *ipv6-address* | **peer-group-name**} **send-community** [**both** | **standard** | **extended**] **Example:** `Device(config-router-af)# neighbor 192.168.2.115 send-community extended`	Specifies that a communities attribute should be sent to the BGP neighbor.
Step 9 **exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode.

Configuring IPv4 and Label iBGP Peering to a Route Reflector

Perform this task to configure IPv4 and label iBGP peering to a route reflector named RR1.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*]
5. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
6. **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4`	Enters address family configuration mode to configure a routing session using standard IPv4 address prefixes.
Step 5	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.115 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 6	**neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label** **Example:** `Device(config-router-af)# neighbor 192.168.2.115 send-label`	Enables label exchange for this address family to this neighbor in order to receive remote PE peer IPv4 loopback with label via RR1 in order to set up an end-to-end LSP.

Configuring the Route Reflector for a Multiautonomous-System Backbone

• Configuring Peering to the PE VPN, page 716

- Configuring the Route Reflector, page 718
- Configuring Peering to the Autonomous System Boundary Router, page 721
- Configuring Peering to Another ISP Route Reflector, page 723

Configuring Peering to the PE VPN

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **address-family vpnv6** [**unicast**]
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
8. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**]
9. **exit**
10. **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*]
11. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
12. **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label**
13. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.

	Command or Action	Purpose
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.115 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table for peering with the route reflector for interautonomous system.
Step 5	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.115 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 6	**address-family vpnv6** [**unicast**] **Example:** `Device(config-router)# address-family vpnv6`	(Optional) Places the device in address family configuration mode.
Step 7	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.115 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 8	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**] **Example:** `Device(config-router-af)# neighbor 192.168.2.115 send-community extended`	Specifies that a community attribute should be sent to the BGP neighbor.
Step 9	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode.

	Command or Action	Purpose
Step 10	**address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4`	Enters address family configuration mode to configure a routing session using standard IPv4 address prefixes.
Step 11	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.115 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 12	**neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label** **Example:** `Device(config-router-af)# neighbor 192.168.2.115 send-label`	Enables label exchange for this address family to this neighbor in order to send to the local PE the remote PE IPv4 loopback with a label in order to set up an end-to-end LSP.
Step 13	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode.

Configuring the Route Reflector

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **address-family vpnv6** [**unicast**]
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
8. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**]
9. **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client**
10. **exit**
11. **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*]
12. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
13. **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label**
14. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.

	Command or Action	Purpose
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.127 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table for peering with the VPN PE for interautonomous system.
Step 5	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.127 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 6	**address-family vpnv6** [**unicast**] **Example:** `Device(config-router)# address-family vpnv6`	(Optional) Places the device in address family configuration mode.
Step 7	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.127 activate`	Enables the exchange of information for this address family with the specified neighbor.
Step 8	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**] **Example:** `Device(config-router-af)# neighbor 192.168.2.127 send-community extended`	Specifies that a community attribute should be sent to the BGP neighbor.
Step 9	**neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client** **Example:** `Device(config-router-af)# neighbor 192.168.2.127 route-reflector-client`	Configures the specified neighbor as a route reflector client.

	Command or Action	Purpose
Step 10	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode.
Step 11	**address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4`	Enters address family configuration mode to configure a routing session using standard IPv4 address prefixes.
Step 12	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.127 activate`	Enables the exchange of information for this address family with the specified neighbor.
Step 13	**neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label** **Example:** `Device(config-router-af)# neighbor 192.168.2.127 send-label`	Enables label exchange for this address family to this neighbor in order to send to the local PE the remote PE IPv4 loopback with a label in order to set up an end-to-end LSP.
Step 14	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode.

Configuring Peering to the Autonomous System Boundary Router

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*]
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
8. **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label**
9. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.102 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table for peering with the ASBR1.

Command or Action	Purpose
Step 5 **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.102 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 6 **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4`	Enters address family configuration mode to configure a routing session using standard IPv4 address prefixes.
Step 7 **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.102 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 8 **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label** **Example:** `Device(config-router-af)# neighbor 192.168.2.102 send-label`	Enables label exchange for this address family to this neighbor in order to receive the remote PE IPv4 loopback with the label set to an end-to-end LSP.
Step 9 **exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode.

Configuring Peering to Another ISP Route Reflector

Perform this task to configure peering to an ISP route reflector named RR2.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **neighbor** {*ip-address* | *ipv6-address*| *peer-group-name*} **ebgp-multihop** [*ttl*]
7. **address-family vpnv6** [**unicast**]
8. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
9. **neighbor** {*ip-address* | *ipv6-address* | **peer-group-name**} **send-community** [**both** | **standard** | **extended**]
10. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **next-hop-unchanged** [**allpaths**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.1 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table for eBGP peering with RR2.

	Command or Action	Purpose
Step 5	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.1 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 6	**neighbor** {*ip-address* | *ipv6-address*| *peer-group-name*} **ebgp-multihop** [*ttl*] **Example:** `Device(config-router)# neighbor 192.168.2.1 ebgp-multihop`	(Optional) Accepts and attempts BGP connections to external peers residing on networks that are not directly connected.
Step 7	**address-family vpnv6** [**unicast**] **Example:** `Device(config-router)# address-family vpnv6`	(Optional) Places the device in address family configuration mode for configuring routing sessions.
Step 8	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.1 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 9	**neighbor** {*ip-address* | *ipv6-address* | **peer-group-name**} **send-community** [**both** | **standard** | **extended**] **Example:** `Device(config-router-af)# neighbor 192.168.2.1 send-community extended`	Specifies that a communities attribute should be sent to the BGP neighbor.
Step 10	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **next-hop-unchanged** [**allpaths**] **Example:** `Device(config-router-af)# neighbor 192.168.2.1 next-hop-unchanged allpaths`	Enables an eBGP multihop peer to propagate to the next hop unchanged for paths.

Configuring the ASBR

Perform this task to configure peering to an ISP route reflector named RR2.

Configuring Peering with Router Reflector RR1

Perform this task to configure peering with a route reflector named RR1.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
8. **neighbor** *ip-address* | *ipv6-address* | *peer-group-name* **send-label**
9. exit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.115 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table for peering with RR1.

Command or Action	Purpose
Step 5 **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.115 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 6 **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name* **Example:** `Device(config-router)# address-family ipv4`	Enters address family configuration mode to configure a routing session using standard IPv4 address prefixes.
Step 7 **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.115 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 8 **neighbor** *ip-address* | *ipv6-address* | *peer-group-name* **send-label** **Example:** `Device(config-router-af)# neighbor 192.168.2.115 send-label`	Enables label exchange for this address family to this neighbor in order to send to the local PE the remote PE IPv4 loopback with a label in order to set up an end-to-end LSP.
Step 9 exit **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode.

Configuring Peering with the Other ISP ASBR2

Perform this task to configure peering with ASBR2.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **neighbor** {*ip-address* | *ipv6-address*| *peer-group-name*} **ebgp-multihop** [*ttl*]
7. **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*]
8. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
9. **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label**
10. **network** {*network-number* [**mask** *network-mask*] | *nsap-prefix*} [**route-map** *map-tag*]
11. **network** {*network-number* [**mask** *network-mask*] | *nsap-prefix*} [**route-map** *map-tag*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.3.1 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table for peering with the ASBR2.

	Command or Action	Purpose
Step 5	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.3.1 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 6	**neighbor** {*ip-address* | *ipv6-address*| *peer-group-name*} **ebgp-multihop** [*ttl*] **Example:** `Device(config-router)# neighbor 192.168.3.1 ebgp-multihop`	Accepts and attempts BGP connections to external peers residing on networks that are not directly connected.
Step 7	**address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4`	Enters address family configuration mode to configure a routing session using standard IPv4 address prefixes.
Step 8	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.3.1 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 9	**neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label** **Example:** `Device(config-router-af)# neighbor 192.168.3.1 send-label`	Enables label exchange for this address family to this neighbor in order to receive the remote PE IPv4 loopback with a label in order to set up an end-to-end LSP.
Step 10	**network** {*network-number* [**mask** *network-mask*] | *nsap-prefix*} [**route-map** *map-tag*] **Example:** `Device(config-router-af)# network 192.168.2.27 mask 255.255.255.255`	Flags a network as local to this autonomous system and enters the network to the BGP table. This configuration is for the PE VPN loopback.

	Command or Action	Purpose
Step 11	**network** {*network-number* [**mask** *network-mask*] | *nsap-prefix*} [**route-map** *map-tag*] **Example:** `Device(config-router-af)# network 192.168.2.15 mask 255.255.255.255`	Flags a network as local to this autonomous system and enters the network to the BGP table. This configuration is for the RR1 loopback.

Configuring CSC for IPv6 VPN

Perform this task to configure CsC-PE1 peering configuration with CsC-CE1.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **hostname** *name*
4. **router bgp** *autonomous-system-number*
5. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast**]
6. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
8. **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**hostname** *name* **Example:** `Device(config)# hostname CSC-PE1`	Specifies or modifies the host name for the network server.

	Command or Action	Purpose
Step 4	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 5	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast**] **Example:** `Device(config-router)# address-family ipv6 vrf ISP2`	Enters address family configuration mode.
Step 6	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router-af)# neighbor FE80::866C:99 Serial0/0 remote-as 200`	Adds an entry to the multiprotocol BGP neighbor table.
Step 7	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor FE80::866C:99 Serial0/0 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 8	**neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label** **Example:** `Device(config-router-af)# neighbor FE80::866C:99 Serial0/0 send-label`	Enables label exchange for this address family to this neighbor.

Configuring BGP IPv6 PIC Edge for IP MPLS

Because many service provider networks contain many VRFs, the BGP PIC feature allows you to configure BGP PIC feature for all VRFs at once. Performing this task in IPv6 address family configuration mode protects IPv6 VRFs.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**]
5. **bgp additional-paths install**
6. **bgp recursion host**

DETAILED STEPS

Command or Action	Purpose
Step 1 **enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2 **configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3 **router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 100`	Configures the BGP routing process.
Step 4 **address-family ipv6** [**vrf** *vrf-name*] [**unicast**	**multicast**	**vpnv6**] **Example:** `Router(config-router)# address-family ipv6 vrf_pic`	Specifies a VRF table named vrf_pic, and enters IPv6 address family configuration mode.
Step 5 **bgp additional-paths install** **Example:** `Router(config-router-af)# bgp additional-paths install`	Calculates a backup path and installs it into the RIB and Cisco Express Forwarding.

	Command or Action	Purpose
Step 6	**bgp recursion host** **Example:** `Router(config-router-af)# bgp recursion host`	Enables the recursive-via-host flag for IPv6 address families.

Verifying and Troubleshooting IPv6 VPN

When users troubleshoot IPv6, any function that works similarly to VPNv4 will likely work for IPv6, therefore minimizing the learning curve for new IPv6 users. Few of the tools and commands used to troubleshoot 6PE and 6VPE are specific to IPv6; rather, the troubleshooting methodology is the same for both IPv4 and IPv6, and the commands and tools often vary by only one keyword.

Verifying and Troubleshooting Routing

Deploying 6PE and 6VPE involves principally BGP. The same set of commands used for VPNv4 can be used (with different set of arguments) for IPv6, and similar outputs are obtained.

Example: BGP IPv6 Activity Summary

```
Device# show bgp ipv6 summary

For address family: IPv6 Unicast
BGP router identifier 192.168.2.126, local AS number 33751
BGP table version is 15, main routing table version 15
12 network entries using 1692 bytes of memory
22 path entries using 1672 bytes of memory
5/4 BGP path/bestpath attribute entries using 580 bytes of memory
14 BGP rrinfo entries using 336 bytes of memory
2 BGP AS-PATH entries using 48 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 4328 total bytes of memory
Dampening enabled. 0 history paths, 0 dampened paths
BGP activity 13/1 prefixes, 23/1 paths, scan interval 60 secs
Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
192.168.2.146     4 33751     991     983       15    0    0 16:26:21       10
192.168.2.147     4 33751     991     983       15    0    0 16:26:22       10
FE80::4F6B:44%Serial1/0
                4 20331     982     987       15    0    0 14:55:52        1
```

Example: Dumping the BGP IPv6 Tables

Each table (for example, BGP IPv6, BGP IPv6 VPN) can be reviewed individually, as shown in the following example:

```
Device# show bgp ipv6 unicast
BGP table version is 15, local router ID is 192.168.2.126
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric    LocPrf Weight Path
* i2001:DB8:100::/48 ::FFFF:192.168.2.101     0    100      0 10000 ?
*>i                  ::FFFF:192.168.2.101     0    100      0 10000 ?
* i2001:DB8::1/128   ::FFFF:192.168.2.101     0    100      0  i
*>i                  ::FFFF:192.168.2.101     0    100      0  i
```

Example: Dumping the IPv6 Routing Tables

IPv6 routing tables identify each routing protocol contributor to routable entries, as shown in the following example:

```
Device# show ipv6 route
IPv6 Routing Table - default - 13 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, R - RIP, I1 - ISIS L1, I2 - ISIS L2
       IA - ISIS interarea, IS - ISIS summary
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
B   2001:DB8:100::/48 [200/0]
     via 192.168.2.101 Default-IP-Routing-Table, indirectly connected
B   2001:DB8::1/128 [200/0]
     via 192.168.2.101 Default-IP-Routing-Table, c
LC  2001:DB8::26/128 [0/0]
     via Loopback0, receive
```

From an IPv6 routing perspective, entries reachable over the MPLS backbone are listed as being indirectly connected, because MPLS is providing a Layer 2 tunnel mechanism.

Verifying and Troubleshooting Forwarding

Forwarding anomalies should be detected and understood so that users can perform troubleshooting. Commands such as **ping ipv6** and **traceroute ipv6** are used to validate data-plane connectivity and detect traffic black-holing. Commands such as **traceroute mpls**and **show mpls forwarding** can pinpoint a damaged node, interface, and forwarding error correction (FEC). At the edge, troubleshooting forwarding failures for a particular IPv6 destination commonly leads to breaking down the recursive resolution into elementary pieces. This task requires combining analysis of IPv6 routing (iBGP or eBGP), IP routing (IS-IS or OSPF), label distribution (BGP, LDP, or RSVP), and adjacency resolution to find a resolution breakage.

The following examples describe how to verify IPv6 VPN and troubleshoot various IPv6 VPN forwarding situations:

- Example: PE-CE Connectivity, page 734
- Example: PE Imposition Path, page 735
- Example: PE Disposition Path, page 737
- Example: Label Switch Path, page 737
- Example: VRF Information, page 738

Example: PE-CE Connectivity

The **ipv6 ping** and **traceroute** commands are useful to check connectivity from a PE to a CE, whether locally attached or remote over the MPLS backbone.

When a device is locally attached, one can use the **ipv6 ping** command with the CE link-local address (used for eBGP peering), as shown in the following example:

```
Device# ping FE80::4F6B:44%Serial1/0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FE80::4F6B:44, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/33/48 ms
```

The **ipv6 ping** command also can be used to test remote PE or CE reachability, but only IPv6 global addresses can be used (link-local addresses are not advertised beyond the link):

```
Device# ping 2001:DB8:1120:1::44
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:1120:1:44::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/33/48 ms
```

The **ping ipv6** and **traceroute** command functions over MPLS require PEs and CEs to announce one IPv6 global prefix. Each 6PE device announces 2001:DB8::PE#/128, filtered at the autonomous system edge. Each IPv6 CE configures 2001:DB8:*prefix*:CE#/128 and announces it as part as its less-specific prefix (2001:DB8:*prefix*::/n).

Reachability of remote PEs and CEs can be tested by using the **traceroute** command. If you have configured all PEs with the **no mpls ip propagate-ttl forwarded** command, when the **traceroute** command is executed from a CE, its output will show only the IPv6 nodes:

```
Device# traceroute 2001:DB8::1
Type escape sequence to abort.
Tracing the route to 2001:DB8::1
  1 2001:DB8::26 [AS 33751] 32 msec 32 msec 20 msec
  2 2001:DB8::1 [AS 33751] [MPLS: Label 73 Exp 0] 20 msec 20 msec 20 msec
  3 2001:DB8::1 [AS 33751] 28 msec 20 msec 20 msec
```

After the P devices have been upgraded with images that support ICMPv6, the **traceroute** command executed on the PE device (Time to Live [TTL] is then propagated) will also show P devices' responses, as shown in the following example:

```
Device# traceroute 2001:DB8::1

Type escape sequence to abort.
Tracing the route to 2001:DB8::1
  1 ::FFFF:172.20.25.1 [MPLS: Labels 38/73 Exp 0] 40 msec 32 msec 32 msec
  2 ::FFFF:172.20.10.1 [MPLS: Labels 30/73 Exp 0] 60 msec 32 msec 32 msec
  3 2001:DB8::1 [MPLS: Label 73 Exp 0] 32 msec 32 msec 16 msec
```

When run from a 6VPE device, both the **ping ipv6** and **traceroute** commands accept a *vrf* argument, exactly as in the case of VPNv4.

The **traceroute** command is useful for evaluating the path across the MPLS backbone, but not for troubleshooting data-plane failures. The P devices are IPv6 unaware (and are also VPNv4 unaware), so the ICMPv6 messages that they generate in response to the **traceroute** command are forwarded to the egress PE using the received label stack. The egress PE can route the ICMPv6 message to the source of the traceroute. When the MPLS path is broken, it is also broken from the ICMP message, which cannot reach the egress PE.

Example: PE Imposition Path

On Cisco devices, the most useful tool for troubleshooting the imposition path for IPv6 is the **show ipv6 cef** command.

Dumping IPv6 Forwarding Table

You can use the **show ipv6 cef** command to display the forwarding table with label stacks used for each destination prefix, as shown in the following example:

```
Device# show ipv6 cef

2001:DB8:100::/48
  nexthop 172.20.25.1 Serial0/0 label 38 72
2001:DB8::1/128
  nexthop 172.20.25.1 Serial0/0 label 38 73
2001:DB8::26/128
  attached to Loopback0, receive
```

Details of an IPv6 Entry in the Forwarding Table

You can use the **show ipv6 cef** command to display details for a specific entry and to analyze how the destination was resolved and the label stack computed, as shown in the following example:

```
Device# show ipv6 cef 2001:DB8:100::/48 internal
2001:DB8:100::/48, epoch 0, RIB[B], refcount 4
  sources: RIB
..
  recursive via 192.168.2.101[IPv4:Default] label 72, fib 0252B1F8, 1 terminal fib
    path 024F56A8, path list 024F0BA8, share 0/1, type attached nexthop
    ifnums: (none)
     path_list contains at least one resolved destination(s). HW IPv4 notified.
    nexthop 172.20.25.1 Serial0/0 label 38, adjacency IP adj out of Serial0/0 0289BEF0
  output chain: label 72 label 38 TAG adj out of Serial0/0 0289BD80
```

Details of a BGP Entry in the BGP Table

The detailed output in the previous example shows that each label composing the label stack has a different origin that can be tracked down individually. The BGP table has the bottom label, as shown in the following example:

```
Device# show bgp ipv6 unicast 2001:DB8:100::/48

BGP routing table entry for 2001:DB8:100::/48, version 2
Paths: (2 available, best #2, table default)
  Advertised to update-groups:
     1
  10000
    ::FFFF:192.168.2.101 (metric 30) from 192.168.2.147 (192.168.2.147)
      Origin incomplete, metric 0, localpref 100, valid, internal
      Originator: 192.168.2.101, Cluster list: 192.168.2.147,
      mpls labels in/out nolabel/72
  10000
    ::FFFF:192.168.2.101 (metric 30) from 192.168.2.146 (192.168.2.146)
      Origin incomplete, metric 0, localpref 100, valid, internal, best
      Originator: 192.168.2.101, Cluster list: 192.168.2.146,
      mpls labels in/out nolabel/72
```

LDP displays the other labels:

```
Device# show mpls ldp bindings 192.168.2.101 32
  lib entry: 192.168.2.101/32, rev 56
        local binding:  label: 40
        remote binding: lsr: 192.168.2.119:0, label: 38
Device# show mpls ldp bindings 172.20.25.0 24
  lib entry: 172.20.25.0/24, rev 2
        local binding: label: imp-null
        remote binding: lsr: 192.168.2.119:0, label: imp-null
```

Example: PE Disposition Path

Use the following examples to troubleshoot the disposition path.

Dumping the MPLS Forwarding Table

The following example illustrates MPLS forwarding table information for troubleshooting the disposition path.

```
Device# show mpls forwarding-table
Local  Outgoing      Prefix            Bytes Label   Outgoing   Next Hop
Label  Label or VC   or Tunnel Id      Switched      interface
16     Pop Label     192.168.2.114/32    0             Se0/0      point2point
17     26            192.168.2.146/32    0             Se0/0      point2point
..
72     No Label      2001:DB8:100::/48     63121         Se1/0      point2point
73     Aggregate     2001:DB8::1/128   24123
```

BGP Label Analysis

The following example illustrates the label used for switching, which has been announced by iBGP (6PE in this example) and can be checked:

```
Device# show bgp ipv6 2001:DB8:100::/48

BGP routing table entry for 2001:DB8:100::/48, version 2
Paths: (1 available, best #1, table default)
  Advertised to update-groups:
     2
  10000
    FE80::2710:2 (FE80::2710:2) from FE80::2710:2%Serial1/0 (192.168.2.103)
      Origin incomplete, metric 0, localpref 100, valid, external, best,
```

Example: Label Switch Path

Because the 6PE and 6VPE LSP endpoints are IPv4 addresses, the IPv4 tools for troubleshooting LSPs are useful for detecting data-plane failures that would lead to IPv6 traffic black-holing.

Analyzing the Label Switch Path

The following example displays the LSP IPv4 end:

```
Device# show ipv6 route 2001:DB8::1/128
Routing entry for 2001:DB8::1/128
  Known via "bgp 33751", distance 200, metric 0, type internal
  Route count is 1/1, share count 0
  Routing paths:
    192.168.2.101%Default-IP-Routing-Table indirectly connected
      MPLS Required
      Last updated 02:42:12 ago
```

Traceroute LSP Example

The following example shows the traceroute LSP:

```
Device# traceroute mpls ipv4 192.168.2.101/32 verbose

Tracing MPLS Label Switched Path to 192.168.2.101/32, timeout is 2 seconds
Codes: '!' - success, 'Q' - request not transmitted,
       '.' - timeout, 'U' - unreachable,
       'R' - downstream router but not target,
       'M' - malformed request
```

```
Type escape sequence to abort.
  0 172.20.25.2 0.0.0.0 MRU 1500 [Labels: 38 Exp: 0]
R 1 172.20.25.1 0.0.0.0 MRU 1500 [Labels: 30 Exp: 0] 40 ms, ret code 6
R 2 172.20.10.1 0.0.0.0 MRU 1504 [Labels: implicit-null Exp: 0] 60 ms, ret code 6
! 3 172.20.40.1 48 ms
```

Example: VRF Information

The following entries show VRF information for 6VPE.

show ipv6 cef vrf

The following is sample output from a Cisco Express Forwarding FIB associated with a VRF named cisco1:

```
Device# show ipv6 cef vrf cisco1
 2001:8::/64
  attached to FastEthernet0/0
 2001:8::3/128
  receive
 2002:8::/64
  nexthop 10.1.1.2 POS4/0 label 22 19
 2010::/64
  nexthop 2001:8::1 FastEthernet0/0
 2012::/64
  attached to Loopback1
 2012::1/128
  receive
```

show ipv6 route vrf

The following is sample output regarding an IPv6 routing table associated with a VRF named cisco1:

```
Device# show ipv6 route vrf cisco1
IPv6 Routing Table cisco1 - 6 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
C   2001:8::/64 [0/0]
     via ::, FastEthernet0/0
L   2001:8::3/128 [0/0]
     via ::, FastEthernet0/0
B   2002:8::/64 [200/0]
     via ::FFFF:192.168.1.4,
B   2010::/64 [20/1]
     via 2001:8::1,
C   2012::/64 [0/0]
     via ::, Loopback1
L   2012::1/128 [0/0]
     via ::, Loopback1
```

Debugging Routing and Forwarding

For troubleshooting of routing and forwarding anomalies, enabling debugging commands can prove useful, although several debug messages can slow the router and harm the usability of such a tool. For this reason, use **debug** commands with caution. The **debug ipv6 cef**, **debug mpls packet**, and **debug ipv6 packet** commands are useful for troubleshooting the forwarding path; the **debug bgp ipv6** and **debug bgp vpnv6** commands are useful for troubleshooting the control plane.

Configuration Examples for Implementing IPv6 VPN over MPLS

Example: IPv6 VPN Configuration Using IPv4 Next Hop

The following example illustrates a 6VPE next hop:

```
interface Loopback0
 ip address 192.168.2.11 255.255.255.255
!
router bgp 100
 neighbor 192.168.2.10 remote-as 100
 neighbor 192.168.2.10 update-source Loopback0
!
 address-family vpnv6
 neighbor 192.168.2.10 activate
 neighbor 192.168.2.10 send-community extended
 exit-address-family
```

By default, the next hop advertised will be the IPv6 VPN address:

```
[0:0]::FFFF:192.168.2.10
```

Note that it is a 192-bit address in the format of [RD]::FFFF:IPv4-address.

When the BGP IPv6 VPN peers share a common subnet, the MP_REACH_NLRI attribute contains a link-local address next hop in addition to the global address next hop. This situation typically occurs in an interautonomous-system topology when ASBRs are facing each other. In that case, the link-local next hop is used locally, and the global next hop is readvertised by BGP.

The BGP next hop is the keystone for building the label stack. The inner label is obtained from the BGP NLRI, and the outer label is the label distribution protocol (LDP) label to reach the IPv4 address embedded into the BGP next hop.

Additional References

Related Documents

Related Topic	Document Title
IPv6 Multiprotocol BGP	Implementing Multiprotocol BGP for IPv6
IPv6 EIGRP	Implementing EIGRP for IPv6
IPv6 MPLS	Implementing IPv6 over MPLS
IPv6 static routes	Implementing Static Routes for IPv6
IPv6 commands	*Cisco IOS IPv6 Command Reference*

Related Topic	Document Title
BGP PIC edge for IP and MPLS-VPN	" BGP PIC Edge for IP and MPLS-VPN ," *IP Routing: BGP Configuration Guide*

Standards

Standard	Title
draft-bonica-internet-icmp	*ICMP Extensions for Multiprotocol Label Switching*
draft-ietf-idr-bgp-ext-communities-0x.txt	*Cooperative Route Filtering Capability for BGP-4*

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 1267	*A Border Gateway Protocol 3 (BGP-3)*
RFC 1772	*Application of the Border Gateway Protocol in the Internet*
RFC 1918	*Address Allocation for Private Internets*
RFC 2858	*Multiprotocol Extensions for BGP-4*
RFC 3107	*Carrying Label Information in BGP-4*
RFC 3392	*Capabilities Advertisement with BGP-4*
RFC 3513	*Internet Protocol Version 6 (IPv6) Addressing Architecture*
RFC 4007	*IPv6 Scoped Address Architecture*
RFC 4193	*Unique Local IPv6 Unicast Addresses*
RFC 4364	*BGP MPLS/IP Virtual Private Networks (VPNs)*
RFC 4382	*MPLS/BGP Layer 3 Virtual Private Network (VPN) Management Information Base*

RFC	Title
RFC 4659	*BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing IPv6 VPN over MPLS

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 31 Feature Information for Implementing IPv6 VPN over MPLS

Feature Name	Releases	Feature Information
BGP IPv6 PIC Edge and Core for IP/MPLS	15.1(2)S	The BGP IPv6 PIC Edge for IP/MPLS feature improves convergence after a network failure. The following commands were modified in this feature: **bgp additional-paths install**, **bgp advertise-best-external**, **bgp recursion host**.
IPv6 VPN over MPLS (6VPE)	12.2(28)SB 12.2(33)SRB 12.2(33)SXI 12.4(20)T 15.0(1)S	The IPv6 VPN (6VPE) over a MPLS IPv4 core infrastructure feature allows ISPs to offer IPv6 VPN services to their customers.

Feature Name	Releases	Feature Information
MPLS VPN 6VPE Support over IP Tunnels	12.2(33)SRB1 12.2(33)SXI	This feature allows the use of IPv4 GRE tunnels to provide IPv6 VPN over MPLS functionality to reach the BGP next hop.

Glossary

- **6VPE device** —Provider edge device providing BGP-MPLS IPv6 VPN service over an IPv4-based MPLS core. It is a IPv6 VPN PE, dual-stack device that implements 6PE concepts on the core-facing interfaces.
- **customer edge (CE) device** —A service provider device that connects to VPN customer sites.
- **Forwarding Information Base (FIB)** —Table containing the information necessary to forward IP datagrams. At a minimum, the FIB contains the interface identifier and next-hop information for each reachable destination network prefix.
- **inbound route filtering (IRF)** —A BGP capability used for filtering incoming BGP updates that are not to be imported by the receiving PE device.
- **IPv6 provider edge device (6PE device)** —Device running a BGP-based mechanism to interconnect IPv6 islands over an MPLS-enabled IPv4 cloud.
- **IPv6 VPN address** —A IPv6 VPN address is a 24-byte identifier, beginning with an 8-byte route distinguisher (RD) and ending with a 16-byte IPv6 address. Sometimes it is called an IPv6 VPN address.
- **IPv6 VPN address family** —The address-family identifier (AFI) identifies a particular network-layer protocol and the subsequent AFI (SAFI) provides additional information. The AFI IPv6 SAFI VPN (AFI=2, SAFI=128) is called the IPv6 VPN address family. Sometimes it is called the IPv6 VPN address family. Similarly AFI IPv4 SAFI VPN is the VPNv4 address family.
- **network layer reachability information (NLRI)** —BGP sends routing update messages containing NLRI to describe a route and how to get there. In this context, an NLRI is a prefix. A BGP update message carries one or more NLRI prefixes and the attributes of a route for the NLRI prefixes; the route attributes include a BGP next hop gateway address and community values.
- **outbound route filtering (ORF)** —A BGP capability used to filtering outgoing BGP routing updates.
- **point of presence (POP)** —Physical location where an interexchange carrier installed equipment to interconnect with a local exchange carrier.
- **provider edge (PE) device** —A service provider device connected to VPN customer sites.
- **route distinguisher (RD)** —A 64-bit value prepended to an IPv6 prefix to create a globally unique IPv6 VPN address.
- **Routing Information Base (RIB)** —Also called the routing table.
- **Virtual routing and forwarding (VRF)** —A VPN routing and forwarding instance in a PE.
- **VRF table** —A routing and a forwarding table associated to a VRF. This is a customer-specific table that enables the PE device to maintain independent routing states for each customer.

Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1110R)

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

Implementing Policy-Based Routing for IPv6

This module describes policy-based routing (PBR) for IPv6. PBR in both IPv6 and IPv4 allows a user to manually configure how received packets should be routed. PBR allows the user to identify packets using several attributes and to specify the next hop or output interface to which the packet should be sent. PBR also provides a basic packet-marking capability.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Implementing Policy-Based Routing for IPv6

The following restrictions for policy-based routing for IPv6 are implemented in Cisco IOS Release 12.2(33)SXI4 on the Cisco Catalyst 6500.

- The **match length** command is applied in software and is not supported in hardware.
- Packet marking is applied in software and is not supported in hardware.
- The **set interface** command is applied in software and is not supported in hardware.
- Packets that contain an IPv6 hop-by-hop header will be supported in software. Such packets will not be supported in hardware.
- PBR policies that use access-lists matching on IPv6 flow label and differentiated services code point (DSCP) value, and extension headers such as routing, mobility, and destination headers, cannot be fully classified in hardware and will be applied in software after partial classification.

- It may not be possible to classify traffic completely in hardware when access-list matching on noncompressible addresses are used. In such cases, PBR is applied in software.
- IPv6 PBR on switch virtual interfaces (SVIs) will be applied in software. Hardware provides only partial classification on SVIs.

Information About Implementing Policy-Based Routing for IPv6

Policy-Based Routing Overview

PBR gives you a flexible means of routing packets by allowing you to configure a defined policy for traffic flows, which lessens reliance on routes derived from routing protocols. To this end, PBR gives you more control over routing by extending and complementing the existing mechanisms provided by routing protocols. PBR allows you to set the IPv6 precedence. It also allows you to specify a path for certain traffic, such as priority traffic over a high-cost link.

PBR for IPv6 may be applied to both forwarded and originated IPv6 packets. For forwarded packets, PBR for IPv6 will be implemented as an IPv6 input interface feature, supported in the process, Cisco Express Forwarding, and distributed Cisco Express Forwarding forwarding paths.

Policies can be based on IPv6 address, port numbers, protocols, or packet size. For a simple policy, you can use any one of these descriptors; for a complex policy, you can use all of them.

PBR allows you to perform the following tasks:

- Classify traffic based on extended access list criteria. Access lists, then, establish the match criteria.
- Set IPv6 precedence bits, giving the network the ability to enable differentiated classes of service.
- Route packets to specific traffic-engineered paths; you might need to route them to allow a specific quality of service (QoS) through the network.

Policies can be based on IPv6 address, port numbers, protocols, or size of packets. For a simple policy, you can use any one of these descriptors; for a complex policy, you can use all of them.

PBR allows you to classify and mark packets at the edge of the network. PBR marks a packet by setting its precedence value. The precedence value can be used directly by routers in the network core to apply the appropriate QoS to a packet, which keeps packet classification at your network edge.

How Policy-Based Routing Works

All packets received on an interface with PBR enabled are passed through enhanced packet filters known as route maps. The route maps used by PBR dictate the policy, determining where to forward packets.

Route maps are composed of statements. The route map statements can be marked as permit or deny, and they are interpreted in the following ways:

- If a packet matches all match statements for a route map that is marked as permit, then the router attempts to policy route the packet using the set statements. Otherwise, the packet is forwarded normally.

- If the packet matches any match statements for a route map that is marked as deny, then the packet is not subject to PBR and is forwarded normally.
- If the statement is marked as permit and the packets do not match any route map statements, the packets are sent back through the normal forwarding channels and destination-based routing is performed.

You specify PBR on the interface that receives the packet, not on the interface from which the packet is sent.

- Packet Matching, page 747
- Packet Forwarding Using Set Statements, page 747

Packet Matching

PBR for IPv6 will match packets using the **match ipv6 address** command in the associated PBR route map. Packet match criteria are those criteria supported by IPv6 access lists, as follows:

- Input interface
- Source IPv6 address (standard or extended access list [ACL])
- Destination IPv6 address (standard or extended ACL)
- Protocol (extended ACL)
- Source port and destination port (extended ACL)
- DSCP (extended ACL)
- Flow-label (extended ACL)
- Fragment (extended ACL)

Packets may also be matched by length using the match length statement in the PBR route map.

Match statements are evaluated first by the criteria specified in the **match ipv6 address** command and then by criteria specified in the **match length** command. Therefore, if both an ACL and a length statement are used, a packet will first be subject to an ACL match. Only packets that pass the ACL match will then be subject to the length match. Finally, only packets that pass both the ACL and the length statement will be policy routed.

Packet Forwarding Using Set Statements

PBR for IPv6 packet forwarding is controlled using a number of set statements in the PBR route map. These set statements are evaluated individually in the order shown, and PBR will attempt to forward the packet using each of the of the set statements in turn. PBR evaluates each set statement by itself, without reference to any prior or subsequent set statement.

You may set multiple forwarding statements in the PBR for IPv6 route map. The following set statements may be specified:

- IPv6 next hop. The next hop to which the packet should be sent. The next hop must be present in the Routing Information Base (RIB), it must be directly connected, and it must be a global IPv6 address. If the next hop is invalid, the set statement is ignored.
- Output interface. A packet is forwarded out of a specified interface. An entry for the packet destination address must exist in the IPv6 RIB, and the specified output interface must be in the path set. If the interface is invalid, the statement is ignored.
- Default IPv6 next hop. The next hop to which the packet should be sent. It must be a global IPv6 address. This set statement is used only when there is no explicit entry for the packet destination in the IPv6 RIB.

- Default output interface. The packet is forwarded out a specified interface. This set statement is used only when there is no explicit entry for the packet destination in the IPv6 RIB.

Note The order in which PBR evaluates the set statements is the order in which they are listed above. This order may differ from the order in which route-map set statements are listed by **show** commands.

When to Use Policy-Based Routing

You might use PBR if you want certain packets to be routed some way other than the obvious shortest path. For example, PBR can be used to provide the following functionality:

- Equal access
- Protocol-sensitive routing
- Source-sensitive routing
- Routing based on interactive versus batch traffic
- Routing based on dedicated links

Some applications or traffic can benefit from QoS-specific routing; for example, you could transfer stock records to a corporate office on a higher-bandwidth, higher-cost link for a short time while sending routine application data such as e-mail over a lower-bandwidth, lower-cost link.

How to Implement Policy-Based Routing for IPv6

Enabling PBR on an Interface

To enable PBR for IPv6, you must create a route map that specifies the packet match criteria and desired policy-route action. Then you associate the route map on the required interface. All packets arriving on the specified interface that match the match clauses will be subject to PBR.

Depending on your release, IPv6 PBR allows users to override normal destination IPv6 address-based routing and forwarding results. Virtual private network (VPN) routing and forwarding (VRF) allows multiple routing instances in Cisco software. The PBR feature is VRF-aware, meaning that it works under multiple routing instances, beyond the default or global routing table.

In PBR, the **set vrf** command decouples the VRF and interface association and allows the selection of a VRF based on ACL-based classification using existing PBR or route-map configurations. It provides a single router with multiple routing tables and the ability to select routes based on ACL classification. The router classifies packets based on ACL, selects a routing table, looks up the destination address, and then routes the packet.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]
4. Do one of the following:
   - **match length** *minimum-length maximum-length*
   -
   - **match ipv6 address** {**prefix-list** *prefix-list-name* | *access-list-name*}
5. Do one of the following:
   - **set ipv6 precedence** *precedence-value*
   -
   - **set ipv6 next-hop** *global-ipv6-address* [*global-ipv6-address...*]
   -
   - set interface type number [...type number]
   -
   - **set ipv6 default next-hop** *global-ipv6-address* [*global-ipv6-address...*]
   -
   - set default interface type number [...type number]
   -
   - **set vrf** *vrf-name*
6. **exit**
7. **interface** *type number*
8. **ipv6 policy route-map** *route-map-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Router(config)# route-map rip-to-ospf permit`	Defines the conditions for redistributing routes from one routing protocol into another, or enables policy routing. • Use the **route-map** command to enter route-map configuration mode.

Step 4	Do one of the following:
	• **match length** *minimum-length maximum-length*
	•
	• **match ipv6 address** {**prefix-list** *prefix-list-name* | *access-list-name*}
	
	Example:
	`Router(config-route-map)# match length 3 200`
	
	Example:
	
	Example:
	`Router(config-route-map)# match ipv6 address marketing`
		Specifies the match criteria.
		• You can specify any or all of the following:
		◦ Matches the Level 3 length of the packet.
		◦ Matches a specified IPv6 access list.
		◦ If you do not specify a **match** command, the route map applies to all packets.

Command or Action	Purpose

Step 5 Do one of the following:
• **set ipv6 precedence** *precedence-value*
•
• **set ipv6 next-hop** *global-ipv6-address* [*global-ipv6-address...*]
•
• set interface type number [...type number]
•
• **set ipv6 default next-hop** *global-ipv6-address* [*global-ipv6-address...*]
•
• set default interface type number [...type number]
•
• **set vrf** *vrf-name*
 	
Example:
`Router(config-route-map)# set ipv6 precedence 1`
 	
Example:
 	
Example:
`Router(config-route-map)# set ipv6 next-hop 2001:DB8:2003:1::95`
 	
Example:
 	
Example:
`Router(config-route-map)# set interface serial 0/0`
 	
Example:
	Specifies the action or actions to take on the packets that match the criteria.
	• You can specify any or all of the following:
	◦ Sets precedence value in the IPv6 header.
	◦ Sets next hop to which to route the packet (the next hop must be adjacent).
	◦ Sets output interface for the packet.
	◦ Sets next hop to which to route the packet, if there is no explicit route for this destination.
	◦ Sets output interface for the packet, if there is no explicit route for this destination.
	◦ Sets VRF instance selection within a route map for a policy-based routing VRF selection.

Command or Action	Purpose

Example:
`Router(config-route-map)# set ipv6 default next-hop 2001:DB8:2003:1::95`
Example:
Example:
`Router(config-route-map)# set default interface ethernet 0`
Example:
Example:
`Router(config-route-map)# set vrf vrfname`	

Step 6 **exit** **Example:** `Router(config-route-map)# exit`	Returns the router to global configuration mode.
Step 7 **interface** *type number* **Example:** `Router(config)# interface FastEthernet 1/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 8 **ipv6 policy route-map** *route-map-name* **Example:** `Router(config-if)# ipv6 policy-route-map interactive`	Identifies a route map to use for IPv6 PBR on an interface.

Enabling Local PBR for IPv6

Packets that are generated by the router are not normally policy routed. Perform this task to enable local PBR for IPv6 for such packets, indicating which route map the router should use.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 local policy route-map** *route-map-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 local policy route-map** *route-map-name* **Example:** `Router(config)# ipv6 local policy route-map pbr-src-90`	Configures PBR for IPv6 for packets generated by the router.

Enabling Cisco Express Forwarding-Switched PBR for IPv6

PBR for IPv6 is supported in the Cisco Express Forwarding switching path. Cisco Express Forwarding-switched PBR is the optimal way to perform PBR on a router.

No special configuration is required to enable Cisco Express Forwarding-switched PBR for IPv6. It is on by default as soon as you enable Cisco Express Forwarding and PBR on the router.

Verifying Configuration and Operation of PBR for IPv6

SUMMARY STEPS

1. **enable**
2. **show ipv6 policy**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ipv6 policy** **Example:** `Router# show ipv6 policy`	Displays IPv6 policy routing packet activity.

Troubleshooting PBR for IPv6

Policy routing looks at various parts of the packet and then routes the packet based on certain user-defined attributes in the packet.

SUMMARY STEPS

1. **enable**
2. **debug ipv6 policy** [*access-list-name*]
3. **show route-map** [*map-name* | **dynamic** *dynamic-map-name* | **application** *application-name*]] | **all**] [**detailed**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**debug ipv6 policy** [*access-list-name*] **Example:** `Router# debug ipv6 policy`	Displays IPv6 policy routing packet activity.

Command or Action	Purpose
Step 3 **show route-map** [*map-name* | **dynamic** *dynamic-map-name* | **application** *application-name*]] | **all**] [**detailed**] **Example:** `Router# show route-map`	Displays all route maps configured or only the one specified.

- Examples, page 755

Examples

Sample Output from the show ipv6 policy Command

The **show ipv6 policy** command displays PBR configuration, as shown in the following example:

```
Router# show ipv6 policy
Interface                Routemap
Ethernet0/0              src-1
```

Sample Output from the show route-map Command

The **show route-map** command displays specific route-map information, such as a count of policy matches:

```
Router# show route-map
route-map bill, permit, sequence 10
  Match clauses:
  Set clauses:
  Policy routing matches:0 packets, 0 bytes
```

Configuration Examples for Implementing Policy-Based Routing for IPv6

- Example: Enabling PBR on an Interface, page 755
- Example: Enabling Local PBR for IPv6, page 756

Example: Enabling PBR on an Interface

In the following example, a route map named pbr-dest-1 is created and configured, specifying packet match criteria and desired policy-route action. PBR is then enabled on Ethernet interface 0/0.

```
ipv6 access-list match-dest-1
  permit ipv6 any 2001:DB8:2001:1760::/32
route-map pbr-dest-1 permit 10
  match ipv6 address match-dest-1
  set interface serial 0/0
interface Ethernet0/0
  ipv6 policy-route-map interactive
```

Example: Enabling Local PBR for IPv6

In the following example, packets with a destination IPv6 address matching that allowed by access list pbr-src-90 are sent to the router at IPv6 address 2001:DB8:2003:1::95:

```
ipv6 access-list src-90
 permit ipv6 host 2001:DB8:2003::90 2001:DB8:2001:1000::/64
route-map pbr-src-90 permit 10
 match ipv6 address src-90
 set ipv6 next-hop 2001:DB8:2003:1::95
ipv6 local policy route-map pbr-src-90
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 addressing and basic configuration	" Implementing IPv6 Addressing and Basic Connectivity ," *Cisco IOS IPv6 Configuration Guide*
QoS for IPv6	" Implementing QoS for IPv6 ," *Cisco IOS IPv6 Configuration Guide*
Multicast Border Gateway Protocol (BGP) for IPv6	" Implementing Multiprotocol BGP for IPv6," *Cisco IOS IPv6 Configuration Guide*
Access control lists for IPv6	" Implementing Traffic Filters and Firewalls for IPv6 Security ," *Cisco IOS IPv6 Configuration Guide*
IPv6 supported feature list	" Start Here: Cisco IOS Software Release Specifics for IPv6 Features ," *Cisco IOS IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*
IPv4 Quality of Service	" Quality of Service Overview ," *Cisco IOS Quality of Service Solutions Configuration Guide*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIBs	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing Policy-Based Routing for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 32 Feature Information for Policy-Based Routing for IPv6

Feature Name	Releases	Feature Information
IPv6 Routing--IPv6 Policy-Based Routing	12.2(30)S 15.1(1)S 12.2(33)SXI4 12.3(7)T 12.4 12.4(2)T	Policy-based routing for IPv6 in Cisco IOS software allows a user to manually configure how received packets should be routed. The following commands were introduced or modified by this feature: **debug ipv6 policy**, **ipv6 local policy route-map**, **ipv6 policy route-map**, **match ipv6 address**, **match length**, **route-map**, **set default interface**, **set interface**, **set ipv6 default next-hop**, **set ipv6 next-hop**, **set ipv6 precedence**, **set vrf**, **show ipv6 policy**, **show route-map**

Implementing QoS for IPv6

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Implementing QoS for IPv6

The following QoS features are not supported for managing IPv6 traffic:

- Compressed Real-Time Protocol (CRTP)
- Network-based application recognition (NBAR)
- Committed access rate (CAR)
- Priority queueing (PQ)
- Custom queueing (CQ)

Information About Implementing QoS for IPv6

Implementation Strategy for QoS for IPv6

IPv6 packets are forwarded by paths that are different from those for IPv4. QoS features supported for IPv6 environments include packet classification, queueing, traffic shaping, weighted random early detection (WRED), class-based packet marking, and policing of IPv6 packets. These features are available at both the process switching and Cisco Express Forwarding switching paths of IPv6.

All of the QoS features available for IPv6 environments are managed from the modular QoS command-line interface (MQC). The MQC allows you to define traffic classes, create and configure traffic policies (policy maps), and then attach those traffic policies to interfaces.

To implement QoS in networks running IPv6, follow the same steps that you would follow to implement QoS in networks running only IPv4. At a very high level, the basic steps for implementing QoS are as follows:

- Know which applications in your network need QoS.
- Understand the characteristics of the applications so that you can make decisions about which QoS features would be appropriate.
- Know your network topology so that you know how link layer header sizes are affected by changes and forwarding.
- Create classes based on the criteria you establish for your network. In particular, if the same network is also carrying IPv4 traffic along with IPv6, decide if you want to treat both of them the same way or treat them separately and specify match criteria accordingly. If you want to treat them the same, use match statements such as **match precedence**, **match dscp**, **set precedence**, and **set dscp**. If you want to treat them separately, add match criteria such as **match protocol ip** and **match protocol ipv6** in a match-all class map.
- Create a policy to mark each class.
- Work from the edge toward the core in applying QoS features.
- Build the policy to treat the traffic.
- Apply the policy.

Packet Classification in IPv6

Packet classification is available with both process and Cisco Express Forwarding switching path. Classification can be based on IPv6 precedence, differentiated services control point (DSCP), and other IPv6 protocol-specific values that can be specified in IPv6 access lists in addition to other non-IPv6 protocol specific values such as COS, packet length, and QOS group. Once you determine which applications need QoS, you can create classes based on the characteristics of the applications. You can use a variety of match criteria to classify traffic. You can combine various match criteria to segregate, isolate, and differentiate traffic.

The enhancements to the modular QoS CLI (MQC) allow you to create matches on precedence, DSCP, and IPv6 access group values in both IPv4 and IPv6 packets. The **match** command allows matches to be made on DSCP values and precedence for both IPv4 and IPv6 packets.

Policies and Class-Based Packet Marking in IPv6 Networks

You can create a policy to mark each class of traffic with appropriate priority values, using either DSCP or precedence. Class-based marking allows you to set the IPv6 precedence and DSCP values for traffic management. The traffic is marked as it enters the router on the ingress interface. The markings are used to

treat the traffic (forward, queue) as it leaves the router on the egress interface. Always mark and treat the traffic as close as possible to its source.

Congestion Management in IPv6 Networks

Once you have marked the traffic, you can use the markings to build a policy and classify traffic on the rest of the network segments. If you keep the policy simple (e.g.,s approximately four classes), it will be easier to manage. Class-based and flow-based queueing are supported for IPv6. The processes and tasks use the same commands and arguments to configure various queueing options for both IP and IPv6.

Congestion Avoidance for IPv6 Traffic

WRED implements the RED-based drop policy on the packets that are likely to overflow the limits of class-based weighted fair queueing (CBWFQ). WRED supports class-based and flow-based (using DSCP or precedence values) queueing.

Traffic Policing in IPv6 Environments

Congestion management for IPv6 is similar to IPv4, and the commands used to configure queueing and traffic shaping features for IPv6 environments are the same commands as those used for IPv4. Traffic shaping allows you to limit the packet dequeue rate by holding additional packets in the queues and forwarding them as specified by parameters configured for traffic shaping features. Traffic shaping uses flow-based queueing by default. CBWFQ can be used to classify and prioritize the packets. Class-based policer and generic traffic shaping (GTS) or Frame Relay traffic shaping (FRTS) can be used for conditioning and policing traffic.

How to Implement QoS for IPv6

Classifying Traffic in IPv6 Networks

The **set cos** and **match cos** commands for 802.1Q (dot1Q) interfaces are supported only for Cisco Express Forwarding-switched packets. Process-switched packets, such as router-generated packets, are not marked when these options are used.

Specifying Marking Criteria for IPv6 Packets

Perform this task to establish the match criteria (or mark the packets) to be used to match packets for classifying network traffic.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy map** *policy-map-name*
4. **class** {*class-name* | **class-default**}
5. Do one of the following:
   - **set precedence** {*precedence-value* | *from-field* [**table** *table-map-name*]}
   - **set** [**ip**] **dscp**{*dscp-value* | *from-field* [**table** *table-map-name*]}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables such as privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy map** *policy-map-name* **Example:** `Router(config)# policy map policy1`	Creates a policy map using the specified name and enters QoS policy-map configuration mode. • Enter name of policy map you want to create.
Step 4	**class** {*class-name*	**class-default**} **Example:** `Router(config-pmap)# class class-default`	Specifies the treatment for traffic of specified class (or the default class) and enters QoS policy-map class configuration mode.

	Command or Action	Purpose

Step 5	Do one of the following:
	• **set precedence** {*precedence-value* | *from-field* [**table** *table-map-name*]}
	• **set** [**ip**] **dscp**{*dscp-value* | *from-field* [**table** *table-map-name*]}
	Example:
	`Router(config-pmap-c)#`
	`set dscp cos table table-map1`
	Example:
	`Router(config-pmap-c)#`
	`set precedence cos table table-map1`
		Sets the precedence value.
		• This example is based on the CoS value (and action) defined in the specified table map.
		• Both precedence and DSCP cannot be changed in the same packets.
		• Sets the DSCP value based on the CoS value (and action) defined in the specified table map.

Using the Match Criteria to Manage IPv6 Traffic Flows

You can use multiple match statements. Depending on the type of class, you can specify whether to match all classes or any of the classes.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map** {*class-name*| **class-default**}
4. Do one of the following:
   - **match precedence** *precedence-value* [*precedence-value precedence-value*]
   - **match access-group name** *ipv6-access-group*
   - **match [ip] dscp** *dscp-value* [*dscp-value dscp-value dscp-value dscp-value dscp-value dscp-value dscp-value*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables such as privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**class-map** {*class-name*| **class-default**} **Example:** `Router(config-pmap-c)# class clsl`	Creates the specified class and enters QoS class-map configuration mode.

Step 4	Do one of the following:
	• **match precedence** *precedence-value* [*precedence-value precedence-value*]
	• **match access-group name** *ipv6-access-group*
	• **match [ip] dscp** *dscp-value* [*dscp-value dscp-value dscp-value dscp-value dscp-value dscp-value dscp-value*
	
	Example:
	`Router(config-pmap-c)#`
	`match precedence 5`
	
	Example:
	`Router(config-pmap-c)# match ip dscp 15`
		Matches the precedence value. The precedence applies to both IPv4 and IPv6 packets.
		or
		Specifies the name of an IPv6 access list against whose contents packets are checked to determine if they belong to the traffic class.
		or
		Identifies a specific IP DSCP value as a match criterion.

Confirming the Service Policy

Ensure that the traffic flow matches the input or output parameter of the policy. For example, downloading a file from an FTP server generates congestion in the receive direction because the server sends large MTU-sized frames, and the client PC returns small acknowledgments (ACKs).

Before you begin this task, simulate congestion with an extended ping using a large ping size and a large number of pings. Also, try downloading a large file from an FTP server. The file constitutes "disturbing" data and fills the interface bandwidth.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number* **multipoint** | **point-to-point**
4. **ip address ip-address mask** [*secondary*]
5. **pvc** [*name*] *vpi* / *vci* [**ces** | **ilmi** | **qsaal** | **smds**]
6. **tx-ring-limit** *ring-limit*
7. **service-policy** {**input** | **output**} *policy-map-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **multipoint**	**point-to-point** **Example:** `Router(config)# interface gigabitethernet1/1/0 point-to-point`	Enters interface configuration mode.
Step 4	**ip address ip-address mask** [*secondary*] **Example:** `Router(config-if)# ip address 10.1.1.1 255.255.255.0`	Specifies the IP address of the interface you want to test.

Step 5	**pvc** [*name*] *vpi* / *vci* [**ces**	**ilmi**	**qsaal**	**smds**]
				
				Example:
				`Router(config-if)#`
				`pvc cisco 0/5`
					Creates or assigns a name to an ATM PVC, optionally specifies the encapsulation type on an ATM PVC, and enters interface-ATM-VC configuration mode.

	Command or Action	Purpose
Step 6	**tx-ring-limit** *ring-limit* **Example:** `Router(config-if-atm-vc)# tx-ring-limit 10`	Reduces the size of the transmit ring of the interface. Lowering this value accelerates the use of the QoS in the Cisco IOS software. • Specify the ring limit as the number of packets for 2600 and 3600 series routers, or as the number of memory particles for 7200 and 7500 series routers.
Step 7	**service-policy** {**input** | **output**} *policy-map-name* **Example:** `Router(config-if-atm-vc)# service-policy output policy9`	Attaches a policy map to an input interface or VC, or an output interface or VC, to be used as the service policy for that interface or VC. • The packets-matched counter is a part of queueing feature and is available only on service policies attached in output direction.

Configuration Examples for Implementing QoS for IPv6

Example: Verifying Cisco Express Forwarding Switching

The following is sample output from the **show cef interface detail** command for Ethernet interface 1/0. Use this command to verify that Cisco Express Forwarding switching is enabled for policy decisions to occur. Notice that the display shows that Cisco Express Forwarding switching is enabled.

```
Router# show cef interface Ethernet 1/0 detail

Ethernet1/0 is up (if_number 9)
  Corresponding hwidb fast_if_number 9
  Corresponding hwidb firstsw->if_number 9
  Internet address is 10.2.61.8/24
  ICMP redirects are always sent
  Per packet load-sharing is disabled
  IP unicast RPF check is disabled
  Inbound access list is not set
  Outbound access list is not set
  IP policy routing is disabled
  Hardware idb is Ethernet1/0
  Fast switching type 1, interface type 5
  IP Distributed CEF switching enabled
  IP Feature Fast switching turbo vector
  IP Feature CEF switching turbo vector
  Input fast flags 0x0, Output fast flags 0x0
  ifindex 7(7)
  Slot 1 Slot unit 0 VC -1
  Transmit limit accumulator 0x48001A82 (0x48001A82)
  IP MTU 1500
```

Example: Verifying Packet Marking Criteria

The following example shows how to use the **match precedence** command to manage IPv6 traffic flows:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
 Router(config)# class-m c1
  Router(config-cmap)# match precedence 5
  Router(config-cmap)# end
Router#
 Router(config)# policy p1
  Router(config-pmap)# class c1
  Router(config-pmap-c)# police 10000 conform set-prec-trans 4
```

To verify that packet marking is working as expected, use the **show policy** command. The output of this command shows a difference in the number of total packets versus the number of packets marked.

```
Router# show policy p1
  Policy Map p1
    Class c1
      police 10000 1500 1500 conform-action set-prec-transmit 4 exceed-action drop
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# interface serial 4/1
Router(config-if)# service out p1
Router(config-if)# end
Router# show policy interface s4/1
 Serial4/1
  Service-policy output: p1
    Class-map: c1 (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: precedence 5
      police:
        10000 bps, 1500 limit, 1500 extended limit
        conformed 0 packets, 0 bytes; action: set-prec-transmit 4
        exceeded 0 packets, 0 bytes; action: drop
        conformed 0 bps, exceed 0 bps violate 0 bps
    Class-map: class-default (match-any)
      10 packets, 1486 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

During periods of transmit congestion at the outgoing interface, packets arrive faster than the interface can send them. It is helpful to know how to interpret the output of the **show policy-map interface** command, which is useful for monitoring the results of a service policy created with Cisco's MQC.

Congestion typically occurs when a fast ingress interface feeds a relatively slow egress interface. Functionally, congestion is defined as filling the transmit ring on the interface (a ring is a special buffer control structure). Every interface supports a pair of rings: a receive ring for receiving packets and a transmit ring for sending packets. The size of the rings varies with the interface controller and with the bandwidth of the interface or virtual circuit (VC). As in the following example, use the **show atm vc** *vcd* command to display the value of the transmit ring on a PA-A3 ATM port adapter.

```
Router# show atm vc 3

ATM5/0.2: VCD: 3, VPI: 2, VCI: 2
VBR-NRT, PeakRate: 30000, Average Rate: 20000, Burst Cells: 94
AAL5-LLC/SNAP, etype:0x0, Flags: 0x20, VCmode: 0x0
OAM frequency: 0 second(s)
PA TxRingLimit: 10
InARP frequency: 15 minutes(s)
Transmit priority 2
InPkts: 0, OutPkts: 0, InBytes: 0, OutBytes: 0
InPRoc: 0, OutPRoc: 0
InFast: 0, OutFast: 0, InAS: 0, OutAS: 0
```

```
InPktDrops: 0, OutPktDrops: 0
CrcErrors: 0, SarTimeOuts: 0, OverSizedSDUs: 0
OAM cells received: 0
OAM cells sent: 0
Status: UP
```

Cisco software (also referred to as the Layer 3 processor) and the interface driver use the transmit ring when moving packets to the physical media. The two processors collaborate in the following way:

- The interface sends packets according to the interface rate or a shaped rate.
- The interface maintains a hardware queue or transmit ring, where it stores the packets waiting for transmission onto the physical wire.
- When the hardware queue or transmit ring fills, the interface provides explicit back pressure to the Layer 3 processor system. It notifies the Layer 3 processor to stop dequeuing packets to the interface's transmit ring because the transmit ring is full. The Layer 3 processor now stores the excess packets in the Layer 3 queues.
- When the interface sends the packets on the transmit ring and empties the ring, it once again has sufficient buffers available to store the packets. It releases the back pressure, and the Layer 3 processor dequeues new packets to the interface.

The most important aspect of this communication system is that the interface recognizes that its transmit ring is full and throttles the receipt of new packets from the Layer 3 processor system. Thus, when the interface is congested, the drop decision is moved from a random, last-in, first-dropped decision in the first in, first out (FIFO) queue of the transmit ring to a differentiated decision based on IP-level service policies implemented by the Layer 3 processor.

Service policies apply only to packets stored in the Layer 3 queues. The table below illustrates which packets sit in the Layer 3 queue. Locally generated packets are always process switched and are delivered first to the Layer 3 queue before being passed on to the interface driver. Fast-switched and Cisco Express Forwarding-switched packets are delivered directly to the transmit ring and sit in the L3 queue only when the transmit ring is full.

Table 33 Packet Types and the Layer 3 Queue

Packet Type	Congestion	Noncongestion
Locally generated packets, including Telnet packets and pings	Yes	Yes
Other packets that are process switched	Yes	Yes
Packets that are Cisco Express Forwarding- or fast-switched	Yes	No

The following example shows these guidelines applied to the **show policy-map interface** command output.

```
Router# show policy-map interface atm 1/0.1

ATM1/0.1: VC 0/100 -
 Service-policy output: cbwfq (1283)
   Class-map: A (match-all) (1285/2)
     28621 packets, 7098008 bytes

     5 minute offered rate 10000 bps, drop rate 0 bps
     Match: access-group 101 (1289)
     Weighted Fair Queueing
       Output Queue: Conversation 73
```

```
    Bandwidth 500 (kbps) Max Threshold 64 (packets)
    (pkts matched/bytes matched) 28621/7098008

    (depth/total drops/no-buffer drops) 0/0/0
Class-map: B (match-all) (1301/4)

  2058 packets, 148176 bytes
  5 minute offered rate 0 bps, drop rate 0 bps
  Match: access-group 103 (1305)
  Weighted Fair Queueing
    Output Queue: Conversation 75
    Bandwidth 50 (kbps) Max Threshold 64 (packets)
    (pkts matched/bytes matched) 0/0
    (depth/total drops/no-buffer drops) 0/0/0
Class-map: class-default (match-any) (1309/0)
  19 packets, 968 bytes
  5 minute offered rate 0 bps, drop rate 0 bps
  Match: any  (1313)
```

The table below defines counters that appear in the example.

Table 34 Packet Counters from show policy-map interface Output

Counter	Explanation
28621 packets, 7098008 bytes	The number of packets matching the criteria of the class. This counter increments whether or not the interface is congested.
(pkts matched/bytes matched) 28621/709800	The number of packets matching the criteria of the class when the interface was congested. In other words, the interface's transmit ring was full, and the driver and the L3 processor system worked together to queue the excess packets in the L3 queues, where the service policy applies. Packets that are process switched always go through the L3 queuing system and therefore increment the "packets matched" counter.
Class-map: B (match-all) (1301/4)	These numbers define an internal ID used with the CISCO-CLASS-BASED-QOS-MIB Management Information Base (MIB).
5 minute offered rate 0 bps, drop rate 0 bps	Use the **load-interval** command to change this value and make it a more instantaneous value. The lowest value is 30 seconds; however, statistics displayed in the **show policy-map interface** command output are updated every 10 seconds. Because the command effectively provides a snapshot at a specific moment, the statistics may not reflect a temporary change in queue size.

Without congestion, there is no need to queue any excess packets. When congestion occurs, packets, including Cisco Express Forwarding- and fast-switched packets, might go into the Layer 3 queue. If you use congestion management features, packets accumulating at an interface are queued until the interface is free to send them; they are then scheduled according to their assigned priority and the queueing mechanism configured for the interface.

Normally, the packets counter is much larger than the packets matched counter. If the values of the two counters are nearly equal, then the interface is receiving a large number of process-switched packets or is heavily congested. Both of these conditions should be investigated to ensure optimal packet forwarding.

Routers allocate conversation numbers for the queues that are created when the service policy is applied. The following example shows the queues and related information.

```
Router# show policy-map interface s1/0.1 dlci 100

      Serial1/0.1: DLCI 100 -
      output : mypolicy
       Class voice
        Weighted Fair Queueing
            Strict Priority
            Output Queue: Conversation 72

              Bandwidth 16 (kbps) Packets Matched 0
             (pkts discards/bytes discards) 0/0
       Class immediate-data
        Weighted Fair Queueing
            Output Queue: Conversation 73

              Bandwidth 60 (%) Packets Matched 0
              (pkts discards/bytes discards/tail drops) 0/0/0
              mean queue depth: 0
              drops: class  random   tail     min-th   max-th   mark-prob
                     0      0        0        64       128      1/10
                     1      0        0        71       128      1/10
                     2      0        0        78       128      1/10
                     3      0        0        85       128      1/10
                     4      0        0        92       128      1/10
                     5      0        0        99       128      1/10
                     6      0        0        106      128      1/10
                     7      0        0        113      128      1/10
                     rsvp   0        0        120      128      1/10
       Class priority-data
        Weighted Fair Queueing
            Output Queue: Conversation 74

              Bandwidth 40 (%) Packets Matched 0 Max Threshold 64 (packets)
              (pkts discards/bytes discards/tail drops) 0/0/0
       Class class-default
        Weighted Fair Queueing
            Flow Based Fair Queueing
            Maximum Number of Hashed Queues 64  Max Threshold 20 (packets)
```

Information reported for each class includes the following:

- Class definition
- Queueing method applied
- Output Queue Conversation number
- Bandwidth used
- Number of packets discarded
- Number of bytes discarded
- Number of packets dropped

The **class-default**class is the default class to which traffic is directed, if that traffic does not satisfy the match criteria of other classes whose policy is defined in the policy map. The **fair-queue** command allows you to specify the number of dynamic queues into which IP flows are sorted and classified. Alternately, routers allocate a default number of queues derived from the bandwidth on the interface or VC. Supported values in either case are a power of two, in a range from 16 to 4096.

The table below lists the default values for interfaces and for ATM permanent virtual circuits (PVCs).

Table 35 Default Number of Dynamic Queues as a Function of Interface Bandwidth

Bandwidth Range	Number of Dynamic Queues
Less than or equal to 64 kbps	16
More than 64 kbps and less than or equal to 128 kbps	32
More than 128 kbps and less than or equal to 256 kbps	64
More than 256 kbps and less than or equal to 512 kbps	128
More than 512 kbps	256

The table below lists the default number of dynamic queues in relation to ATM PVC bandwidth.

Table 36 Default Number of Dynamic Queues as a Function of ATM PVC Bandwidth

Bandwidth Range	Number of Dynamic Queues
Less than or equal to 128 kbps	16
More than 128 kbps and less than or equal to 512 kbps	32
More than 512 kbps and less than or equal to 2000 kbps	64
More than 2000 kbps and less than or equal to 8000 kbps	128
More than 8000 kbps	256

Based on the number of reserved queues for WFQ, Cisco software assigns a conversation or queue number as shown in the table below.

Table 37 Conversation Numbers Assigned to Queues

Number	Type of Traffic
1 to 256	General flow-based traffic queues. Traffic that does not match to a user-created class will match to class-default and one of the flow-based queues.
257 to 263	Reserved for Cisco Discovery Protocol and for packets marked with an internal high-priority flag.

Number	Type of Traffic
264	Reserved queue for the priority class (classes configured with the priority command). Look for the "Strict Priority" value for the class in the **show policy-map** interface output. The priority queue uses a conversation ID equal to the number of dynamic queues, plus 8.
265 and higher	Queues for user-created classes.

Example: Matching DSCP Value

The following example shows how to configure the service policy called priority50 and attach service policy priority50 to an interface. In this example, the **match dscp** command includes the optional **ip** keyword, meaning that the match is for IPv4 packets only. The class map called ipdscp15 will evaluate all packets entering interface Fast Ethernet 1/0. If the packet is an IPv4 packet and has a DSCP value of 15, the packet will be treated as priority traffic and will be allocated with bandwidth of 50 kbps.

```
Router(config)#
 class-map ipdscp15
Router(config-cmap)#
 match ip dscp 15
Router(config)#
 exit
Router(config)#
policy-map priority50
Router(config-pmap)#
 class ipdscp15
Router(config-pmap-c)#
priority 50
Router(config-pmap-c)#
exit
Router(config-pmap)#
 exit
Router(config)#
interface fa1/0
Router(config-if)#
service-policy input priority55
```

To match on IPv6 packets only, use the **match dscp** command without the **ip** keyword preceded by the **match protocol** command. Ensure that the class map has the **match-all** attribute (which is the default).

```
Router(config)#
 class-map ipdscp15
Router(config-cmap)#
 match protocol ipv6
Router(config-cmap)#
 match dscp 15
Router(config)#
 exit
```

To match packets on both IPv4 and IPv6 protocols, use the **match dscp** command:

```
Router(config)#
 class-map ipdscp15
Router(config-cmap)#
 match dscp 15
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 supported feature list	"Start Here: Cisco IOS Software Release Specifics for IPv6 Features ," *Cisco IOS IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*

Standards

Standard	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2474	*Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*
RFC 2475	*An Architecture for Differentiated Services Framework*
RFC 2597	*Assured Forwarding PHB*
RFC 2598	*An Expedited Forwarding PHB*
RFC 2640	*Internet Protocol, Version 6 Specification*
RFC 2697	*A Single Rate Three Color Marker*

RFC	Title
RFC 2698	*A Two Rate Three Color Marker*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing QoS for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 38 Feature Information for Implementing QoS for IPv6

Feature Name	Releases	Feature Information
IPv6 Quality of Service (QoS)	12.0(28)S[1] 12.2(33)SRA 12.2(18)SXE[2] 12.2(13)T 12.3 12.3(2)T 12.4 12.4(2)T	QoS features supported for IPv6 environments include packet classification, queueing, traffic shaping, WRED, class-based packet marking, and policing of IPv6 packets.
IPv6 QoS--MQC Packet Marking/Re-marking	12.0(28)S 12.2(33)SRA 12.2(18)SXE 12.2(13)T 12.3 12.3(2)T 12.4 12.4(2)T	Class-based marking allows you to set the IPv6 precedence and DSCP values for traffic management.

1 Feature is supported on Cisco 12000 series Internet routers in Cisco IOS Release 12.0(28)S.

2 Cisco IOS Release 12.2(18)SXE provides support for this feature. Cisco IOS Release 12.2(18)SXE is specific to Cisco Catalyst 6500 and Cisco 7600 series routers.

Feature Name	Releases	Feature Information
IPv6 QoS--MQC Packet Classification	12.2(33)SRA 12.2(18)SXE 12.2(13)T 12.3 12.3(2)T 12.4 12.4(2)T	The modular QoS CLI allows you to define traffic classes, create and configure traffic policies (policy maps), and then attach those traffic policies to interfaces.
IPv6 QoS--MQC Traffic Policing	12.0(28)S 12.2(33)SRA 12.2(18)SXE 12.2(13)T 12.3 12.3(2)T 12.4 12.4(2)T	Configuration or command usage for policing are the same in IPv6 environments as for IPv4 environments.
IPv6 QoS--MQC Traffic Shaping	12.0(28)S 12.2(33)SRA 12.2(18)SXE 12.2(13)T 12.3 12.3(2)T 12.4 12.4(2)T	Traffic shaping allows you to limit the packet dequeue rate by holding additional packets in the queues and forwarding them as specified by parameters configured for traffic shaping features.
IPv6 QoS--MQC WRED-Based Drop	12.0(28)S 12.2(33)SRA 12.2(18)SXE 12.2(13)T 12.3 12.3(2)T 12.4 12.4(2)T	WRED implements the RED-based drop policy on the packets that are likely to overflow the limits of CBWFQ.
IPv6 QoS--Queueing	12.2(33)SRA 12.2(18)SXE 12.2(13)T 12.3 12.3(2)T 12.4 12.4(2)T	Class-based and flow-based queueing are supported for IPv6.

Implementing RIP for IPv6

This module describes how to configure Routing Information Protocol for IPv6. RIP is a distance-vector routing protocol that uses hop count as a routing metric. RIP is an Interior Gateway Protocol (IGP) most commonly used in smaller networks.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Implementing RIP for IPv6

RIP for IPv6

IPv6 RIP functions the same and offers the same benefits as RIP in IPv4. RIP enhancements for IPv6, detailed in RFC 2080, include support for IPv6 addresses and prefixes, and the use of the all-RIP-routers multicast group address FF02::9 as the destination address for RIP update messages.

In the Cisco software implementation of IPv6 RIP, each IPv6 RIP process maintains a local routing table, referred to as a Routing Information Database (RIB). The IPv6 RIP RIB contains a set of best-cost IPv6 RIP routes learned from all its neighboring networking devices. If IPv6 RIP learns the same route from two different neighbors, but with different costs, it will store only the lowest cost route in the local RIB. The RIB also stores any expired routes that the RIP process is advertising to its neighbors running RIP. IPv6

RIP will try to insert every non-expired route from its local RIB into the master IPv6 RIB. If the same route has been learned from a different routing protocol with a better administrative distance than IPv6 RIP, the RIP route will not be added to the IPv6 RIB but the RIP route will still exist in the IPv6 RIP RIB.

Nonstop Forwarding for IPv6 RIP

Cisco nonstop forwarding (NSF) continues forwarding packets while routing protocols converge, therefore avoiding a route flap on switchover. When an RP failover occurs, the Forwarding Information Base (FIB) marks installed paths as stale by setting a new epoch. Subsequently, the routing protocols reconverge and populate the RIB and FIB. Once all NSF routing protocols converge, any stale routes held in the FIB are removed. A failsafe timer is required to delete stale routes, in case of routing protocol failure to repopulate the RIB and FIB.

RIP registers as an IPv6 NSF client. Doing so has the benefit of using RIP routes installed in the Cisco Express Forwarding table until RIP has converged on the standby.

How to Implement RIP for IPv6

Enabling the IPv6 RIP Process

Before configuring the router to run IPv6 RIP, globally enable IPv6 using the **ipv6 unicast-routing** command in global configuration mode, and enable IPv6 on any interfaces on which IPv6 RIP is to be enabled.

If you want to set or change a global value, follow steps 1 and 2, and then use the optional **ipv6 router rip**command in global configuration mode.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **interface** *type number*
5. **ipv6 rip** *name* **enable**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Router(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.
Step 4	**interface** *type number* **Example:** `Router(config)# interface Ethernet 0/0`	Specifies the interface type and number, and enters interface configuration mode.
Step 5	**ipv6 rip** *name* **enable** **Example:** `Router(config-if)# ipv6 rip process1 enable`	Enables the specified IPv6 RIP routing process on an interface.

Customizing IPv6 RIP

Perform this optional task to customize IPv6 RIP by configuring the maximum numbers of equal-cost paths that IPv6 RIP will support, adjusting the IPv6 RIP timers, and originating a default IPv6 route.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router rip** *word*
4. **maximum-paths** *number-paths*
5. **exit**
6. **interface** *type number*
7. **ipv6 rip** *name* **default-information** {**only** | **originate**} [**metric** *metric-value*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router rip** *word* **Example:** `Router(config)# ipv6 router rip process1`	Configures an IPv6 RIP routing process and enters router configuration mode for the IPv6 RIP routing process. • Use the *word*argument to identify a specific IPv6 RIP routing process.
Step 4	**maximum-paths** *number-paths* **Example:** `Router(config-router)# maximum-paths 1`	(Optional) Defines the maximum number of equal-cost routes that IPv6 RIP can support. • The *number-paths* argument is an integer from 1 to 64. The default for RIP is four paths.
Step 5	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.

	Command or Action	Purpose
Step 6	**interface** *type number* **Example:** `Router(config)# interface Ethernet 0/0`	Specifies the interface type and number, and enters interface configuration mode.
Step 7	**ipv6 rip** *name* **default-information** {**only** | **originate**} [**metric** *metric-value*] **Example:** `Router(config-if)# ipv6 rip process1 default-information originate`	(Optional) Originates the IPv6 default route (::/0) into the specified RIP routing process updates sent out of the specified interface. **Note** To avoid routing loops after the IPv6 default route (::/0) is originated out of any interface, the routing process ignores all default routes received on any interface. • Specifying the **only** keyword originates the default route (::/0) but suppresses all other routes in the updates sent on this interface. • Specifying the **originate** keyword originates the default route (::/0) in addition to all other routes in the updates sent on this interface.

Redistributing Routes into an IPv6 RIP Routing Process

RIP supports the use of a route map to select routes for redistribution. Routes may be specified by prefix, using a route-map prefix list, or by tag, using the route-map "match tag" function.

The maximum metric that RIP can advertise is 16, and a metric of 16 denotes a route that is unreachable. Therefore, if you are redistributing routes with metrics greater than or equal to 16, then by default RIP will advertise them as unreachable. These routes will not be used by neighboring routers. The user must configure a redistribution metric of less than 15 for these routes.

Note You must to advertise a route with metric of 15 or less. A RIP router always adds an interface cost--the default is 1--onto the metric of a received route. If you advertise a route with metric 15, your neighbor will add 1 to it, making a metric of 16. Because a metric of 16 is unreachable, your neighbor will not install the route in the routing table.

If no metric is specified, then the current metric of the route is used. To find the current metric of the route, enter the **show ipv6 route** command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 rip** *word* **enable**
5. **redistribute** *protocol* [*process-id*] {**level-1** | **level-1-2**| **level-2**} [**metric** *metric-value*] [**metric-type**{**internal** | **external**}] [**route-map** *map-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface Ethernet 0/0`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**ipv6 rip** *word* **enable** **Example:** `Router(config-if)# ipv6 router one enable`	Enables an IPv6 Routing Information Protocol (RIP) routing process on an interface.
Step 5	**redistribute** *protocol* [*process-id*] {**level-1** | **level-1-2**| **level-2**} [**metric** *metric-value*] [**metric-type**{**internal** | **external**}] [**route-map** *map-name*] **Example:** `Router(config-router)# redistribute bgp 65001 route-map bgp-to-rip`	Redistributes the specified routes into the IPv6 RIP routing process. • The *protocol* argument can be one of the following keywords: **bgp**, **connected**, **isis**, **rip**, or **static**. • The **rip** keyword and *process-id* argument specify an IPv6 RIP routing process. **Note** The **connected** keyword refers to routes that are established automatically by assigning IPv6 addresses to an interface.

Configuring Route Tags for IPv6 RIP Routes

When performing route redistribution, you can associate a numeric tag with a route. The tag is advertised with the route by RIP and will be installed along with the route in neighboring router's routing table.

If you redistribute a tagged route (for example, a route in the IPv6 routing table that already has a tag) into RIP, then RIP will automatically advertise the tag with the route. If you use a redistribution route map to specify a tag, then RIP will use the route map tag in preference to the routing table tag.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]
4. **match ipv6 address** {**prefix-list** *prefix-list-name* | *access-list-name*
5. **set tag** *tag-value*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**route-map** *map-tag* [**permit**	**deny**] [*sequence-number*] **Example:** `Router(config)# route-map bgp-to-rip permit 10`	Defines a route map, and enters route-map configuration mode. • Follow this step with a **match** command.
Step 4	**match ipv6 address** {**prefix-list** *prefix-list-name*	*access-list-name* **Example:** `Router(config-route-map)# match ipv6 address prefix-list bgp-to-rip-flt`	Specifies a list of IPv6 prefixes to be matched.
Step 5	**set tag** *tag-value* **Example:** `Router(config-route-map)# set tag 4`	Sets the tag value to associate with the redistributed routes.

Filtering IPv6 RIP Routing Updates

Route filtering using distribute lists provides control over the routes RIP receives and advertises. This control may be exercised globally or per interface.

Filtering is controlled by IPv6 distribute lists. Input distribute lists control route reception, and input filtering is applied to advertisements received from neighbors. Only those routes that pass input filtering

will be inserted in the RIP local routing table and become candidates for insertion into the IPv6 routing table.

Output distribute lists control route advertisement; Output filtering is applied to route advertisements sent to neighbors. Only those routes passing output filtering will be advertised.

Global distribute lists (which are distribute lists that do not apply to a specified interface) apply to all interfaces. If a distribute list specifies an interface, then that distribute list applies only to that interface.

An interface distribute list always takes precedence. For example, for a route received at an interface, with the interface filter set to deny, and the global filter set to permit, the route is blocked, the interface filter is passed, the global filter is blocked, and the route is passed.

IPv6 prefix lists are used to specify certain prefixes or a range of prefixes that must be matched before a permit or deny statement can be applied. Two operand keywords can be used to designate a range of prefix lengths to be matched. A prefix length of less than, or equal to, a value is configured with the **le**keyword. A prefix length greater than, or equal to, a value is specified using the **ge** keyword. The **ge** and **le** keywords can be used to specify the range of the prefix length to be matched in more detail than the usual *ipv6-prefix* / *prefix-length* argument. For a candidate prefix to match against a prefix list entry three conditions can exist:

- The candidate prefix must match the specified prefix list and prefix length entry.
- The value of the optional **le** keyword specifies the range of allowed prefix lengths from the *prefix-length* argument up to, and including, the value of the **le** keyword.
- The value of the optional **ge** keyword specifies the range of allowed prefix lengths from the value of the **ge** keyword up to, and including, 128.

Note The first condition must match before the other conditions take effect.

An exact match is assumed when the **ge** or **le** keywords are not specified. If only one keyword operand is specified then the condition for that keyword is applied, and the other condition is not applied. The *prefix-length* value must be less than the **ge** value. The **ge** value must be less than, or equal to, the **le** value. The **le** value must be less than or equal to 128.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 prefix list** *prefix-list-name* **seq** *seq-number*] {**deny** *ipv6-prefix* / *prefix-length* | **description** *text*} [**ge** *ge-value*] [**le** *le-value*
4. **ipv6 prefix list** *prefix-list-name* **seq** *seq-number*] {**deny** *ipv6-prefix* / *prefix-length* | **description** *text*} [**ge** *ge-value*] [**le** *le-value*
5. Repeat Steps 3 and 4 as many times as necessary to build the prefix list.
6. **ipv6 router rip** *name*
7. **distribute-list prefix-list** *prefix-list-name* **in** | **out**} [*interface-type interface-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 prefix list** *prefix-list-name* **seq** *seq-number*] {**deny** *ipv6-prefix* / *prefix-length* | **description** *text*} [**ge** *ge-value*] [**le** *le-value* **Example:** `Router(config)# ipv6 prefix-list abc permit 2001:DB8::/16`	Creates an entry in the IPv6 prefix list.
Step 4	**ipv6 prefix list** *prefix-list-name* **seq** *seq-number*] {**deny** *ipv6-prefix* / *prefix-length* | **description** *text*} [**ge** *ge-value*] [**le** *le-value* **Example:** `Router(config)# ipv6 prefix-list abc deny ::/0`	Creates an entry in the IPv6 prefix list.
Step 5	Repeat Steps 3 and 4 as many times as necessary to build the prefix list.	--
Step 6	**ipv6 router rip** *name* **Example:** `Router(config)# ipv6 router rip process1`	Configures an IPv6 RIP routing process.
Step 7	**distribute-list prefix-list** *prefix-list-name* **in** | **out**} [*interface-type interface-number* **Example:** `Router(config-rtr-rip)# distribute-list prefix-list process1 in ethernet 0/0`	Applies a prefix list to IPv6 RIP routing updates that are received or sent on an interface.

Verifying IPv6 RIP Configuration and Operation

SUMMARY STEPS

1. **show ipv6 rip** [*name*][**database**| **next-hops**]
2. **show ipv6 route** [*ipv6-address*| *ipv6-prefix*/*prefix-length*| *protocol* | *interface-type interface-number*]
3. **enable**
4. **debug ipv6 rip** [*interface-type interface-number*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show ipv6 rip** [*name*][**database**| **next-hops**] **Example:** `Router> show ipv6 rip process1 database`	(Optional) Displays information about current IPv6 RIP processes. • In this example, IPv6 RIP process database information is displayed for the specified IPv6 RIP process.
Step 2	**show ipv6 route** [*ipv6-address*| *ipv6-prefix*/*prefix-length*| *protocol*	*interface-type interface-number*] **Example:** `Router> show ipv6 route rip`	(Optional) Displays the current contents of the IPv6 routing table. • In this example, only IPv6 RIP routes are displayed.
Step 3	**enable** **Example:** `Router> enable`	Enables higher privilege levels, such as privileged EXEC mode. • Enter your password if prompted.
Step 4	**debug ipv6 rip** [*interface-type interface-number*] **Example:** `Router# debug ipv6 rip`	(Optional) Displays debugging messages for IPv6 RIP routing transactions.

• Examples, page 786

Examples

Sample Output from the show ipv6 rip Command

In the following example, output information about all current IPv6 RIP processes is displayed using the **show ipv6 rip**command:

```
Router> show ipv6 rip
```

```
RIP process "process1", port 521, multicast-group FF02::9, pid 62
     Administrative distance is 120. Maximum paths is 1
     Updates every 5 seconds, expire after 15
     Holddown lasts 10 seconds, garbage collect after 30
     Split horizon is on; poison reverse is off
     Default routes are generated
     Periodic updates 223, trigger updates 1
  Interfaces:
    Ethernet0/0
  Redistribution:
    Redistributing protocol bgp 65001 route-map bgp-to-rip
```

In the following example, output information about a specified IPv6 RIP process database is displayed using the **show ipv6 rip**command with the *name* argument and the **database** keyword. In the following output for the IPv6 RIP process named process1, timer information is displayed, and route 2001:DB8::16/64 has a route tag set:

```
Router> show ipv6 rip process1 database
RIP process "process1", local RIB
 2001:DB8::/64, metric 2
     Ethernet0/0/FE80::A8BB:CCFF:FE00:B00, expires in 13 secs
 2001:DB8::/16, metric 2 tag 4, installed
     Ethernet0/0/FE80::A8BB:CCFF:FE00:B00, expires in 13 secs
 2001:DB8:1::/16, metric 2 tag 4, installed
     Ethernet0/0/FE80::A8BB:CCFF:FE00:B00, expires in 13 secs
 2001:DB8:2::/16, metric 2 tag 4, installed
     Ethernet0/0/FE80::A8BB:CCFF:FE00:B00, expires in 13 secs
 ::/0, metric 2, installed
     Ethernet0/0/FE80::A8BB:CCFF:FE00:B00, expires in 13 secs
```

In the following example, output information for a specified IPv6 RIP process is displayed using the **show ipv6 rip**user EXEC command with the *name* argument and the **next-hops** keyword:

```
Router> show ipv6 rip process1 next-hops
RIP process "process1", Next Hops
  FE80::A8BB:CCFF:FE00:A00/Ethernet0/0 [4 paths]
```

Sample Output from the show ipv6 route Command

The current metric of the route can be found by entering the **show ipv6 route** command. In the following example, output information for all IPv6 RIP routes is displayed using the **show ipv6 route**command with the **rip** protocol keyword:

```
Router> show ipv6 route rip
IPv6 Routing Table - 17 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
R   2001:DB8:1::/32 [120/2]
     via FE80::A8BB:CCFF:FE00:A00, Ethernet0/0
R   2001:DB8:2::/32 [120/2]
     via FE80::A8BB:CCFF:FE00:A00, Ethernet0/0
R   2001:DB8:3::/32 [120/2]
     via FE80::A8BB:CCFF:FE00:A00, Ethernet0/0
```

Sample Output from the debug ipv6 rip Command

In the following example, debugging messages for IPv6 RIP routing transactions are displayed using the **debug ipv6 rip**command:

Note By default, the system sends the output from **debug** commands and system error messages to the console. To redirect debugging output, use the **logging** command options within privileged EXEC mode. Possible destinations include the console, virtual terminals, internal buffer, and UNIX hosts running a syslog server.

```
Router# debug ipv6 rip
RIPng: Sending multicast update on Ethernet0/0 for process1
       src=FE80::A8BB:CCFF:FE00:B00
       dst=FF02::9 (Ethernet0/0)
       sport=521, dport=521, length=112
       command=2, version=1, mbz=0, #rte=5
       tag=0, metric=1, prefix=2001:DB8::/64
       tag=4, metric=1, prefix=2001:DB8:1::/16
       tag=4, metric=1, prefix=2001:DB8:2;:/16
       tag=4, metric=1, prefix=2001:DB8:3::/16
       tag=0, metric=1, prefix=::/0
RIPng: Next RIB walk in 10032
RIPng: response received from FE80::A8BB:CCFF:FE00:A00 on Ethernet0/0 for process1
       src=FE80::A8BB:CCFF:FE00:A00 (Ethernet0/0)
       dst=FF02::9
       sport=521, dport=521, length=92
       command=2, version=1, mbz=0, #rte=4
       tag=0, metric=1, prefix=2001:DB8::/64
       tag=0, metric=1, prefix=2001:DB8:1::/32
       tag=0, metric=1, prefix=2001:DB8:2::/32
       tag=0, metric=1, prefix=2001:DB8:3::/32
```

Configuration Examples for IPv6 RIP

Example IPv6 RIP Configuration

In the following example, the IPv6 RIP process named process1 is enabled on the router and on Ethernet interface 0/0. The IPv6 default route (::/0) is advertised in addition to all other routes in router updates sent on Ethernet interface 0/0. Additionally, BGP routes are redistributed into the RIP process named process1 according to a route map where routes that match a prefix list are also tagged. The number of parallel paths is set to one to allow the route tagging, and the IPv6 RIP timers are adjusted. A prefix list named eth0/0-in-flt filters inbound routing updates on Ethernet interface 0/0.

```
ipv6 router rip process1
 maximum-paths 1
 redistribute bgp 65001 route-map bgp-to-rip
 distribute-list prefix-list eth0/0-in-flt in Ethernet0/0
!
interface Ethernet0/0
 ipv6 address 2001:DB8::/64 eui-64
 ipv6 rip process1 enable
 ipv6 rip process1 default-information originate
!
ipv6 prefix-list bgp-to-rip-flt seq 10 deny 2001:DB8:3::/16 le 128
ipv6 prefix-list bgp-to-rip-flt seq 20 permit 2001:DB8:1::/8 le 128
!
ipv6 prefix-list eth0/0-in-flt seq 10 deny ::/0
ipv6 prefix-list eth0/0-in-flt seq 15 permit ::/0 le 128
!
route-map bgp-to-rip permit 10
 match ipv6 address prefix-list bgp-to-rip-flt
 set tag 4
```

Additional References

Related Documents

Related Topic	Document Title
IPv4 RIP configuration tasks	" Configuring Routing Information Protocol ," *Cisco IOS IP Routing Protocols Configuration Guide*
RIP commands: complete command syntax, command mode, defaults, usage guidelines, and examples	" RIP Commands ," *Cisco IOS IP Routing Protocols Command Reference*
IPv6 supported feature list	" Start Here: Cisco IOS Software Release Specifics for IPv6 Features ," *Cisco IOS IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIBs	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 2080	*RIPng for IPv6*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing RIP for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 39 Feature Information for Implementing RIP for IPv6

Feature Name	Releases	Feature Information
IPv6--RIPng Nonstop Forwarding	12.2(33)SRE 15.0(1)SY	The IPv6 RIPng nonstop forwarding feature is supported.
IPv6 Routing--RIP for IPv6 (RIPng)	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	RIP enhancements for IPv6 include support for IPv6 addresses and prefixes, and the use of the all-RIP-routers multicast group address FF02::9 as the destination address for RIP update messages.
IPv6 Routing--Route Redistribution	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T	Routes may be specified by prefix, using a route-map prefix list, or by tag, using the route-map "match tag" function.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

Implementing Traffic Filters and Firewalls for IPv6 Security

This module describes how to configure Cisco IOS IPv6 traffic filter and firewall features for your Cisco networking devices. These security features can protect your network from degradation or failure and also from data loss or compromised security resulting from intentional attacks and from unintended but damaging mistakes by well-meaning network users.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Implementing Traffic Filters and Firewalls for IPv6 Security

Cisco IOS Release 12.2(2)T through Cisco IOS Release 12.2(13)T and Cisco IOS Release 12.0(22)S and later releases support only standard IPv6 access control list (ACL) functionality. In Cisco IOS Release 12.0(23)S and 12.2(13)T or later releases, the standard IPv6 ACL functionality is extended to support traffic filtering based on IPv6 option headers and optional, upper-layer protocol type information for finer granularity of control (functionality similar to extended ACLs in IPv4).

Information About Implementing Traffic Filters and Firewalls for IPv6 Security

Access Control Lists for IPv6 Traffic Filtering

The standard ACL functionality in IPv6 is similar to standard ACLs in IPv4. Access lists determine what traffic is blocked and what traffic is forwarded at router interfaces and allow filtering based on source and destination addresses, inbound and outbound to a specific interface. Each access list has an implicit deny statement at the end. IPv6 ACLs are defined and their deny and permit conditions are set using the **ipv6 access-list**command with the **deny** and **permit** keywords in global configuration mode.

IPv6 extended ACLs augments standard IPv6 ACL functionality to support traffic filtering based on IPv6 option headers and optional, upper-layer protocol type information for finer granularity of control (functionality similar to extended ACLs in IPv4).

IPv6 ACL Extensions for IPsec Authentication Header

This feature provides the ability to match on the upper layer protocol (ULP) (for example, TCP, User Datagram Protocol [UDP], ICMP, SCTP) regardless of whether an authentication header (AH) is present or absent.

TCP or UDP traffic can be matched to the upper-layer protocol (ULP) (for example, TCP, UDP, ICMP, SCTP) if an AH is present or absent. Before this feature was introduced, this function was only available if an AH was absent.

This feature introduces the keyword **auth** to the **permit**and **deny**commands. The **auth** keyword allows matching traffic against the presence of the authentication header in combination with the specified protocol; that is, TCP or UDP.

IPv6 traffic can be matched to a ULP when an AH header is present. To perform this function, enter the **ahp** option for the *protocol* argument when using the **permit** or **deny** command.

Access Class Filtering in IPv6

Filtering incoming and outgoing connections to and from the router based on an IPv6 ACL is performed using the **ipv6 access-class** command in line configuration mode. The **ipv6 access-class** command is similar to the **access-class** command, except the IPv6 ACLs are defined by a name. If the IPv6 ACL is applied to inbound traffic, the source address in the ACL is matched against the incoming connection source address and the destination address in the ACL is matched against the local router address on the interface. If the IPv6 ACL is applied to outbound traffic, the source address in the ACL is matched against

the local router address on the interface and the destination address in the ACL is matched against the outgoing connection source address. We recommend that identical restrictions are set on all the virtual terminal lines because a user can attempt to connect to any of them.

Tunneling Support

IPv6 packets tunneled in IPv4 are not inspected. If a tunnel terminates on a router, and IPv6 traffic exiting the tunnel is nonterminating, then the traffic is inspected.

Virtual Fragment Reassembly

When VFR is enabled, VFR processing begins after ACL input lists are checked against incoming packets. The incoming packets are tagged with the appropriate VFR information.

Cisco IOS Firewall for IPv6

The Cisco IOS Firewall feature provides advanced traffic filtering functionality as an integral part of a network's firewall. Cisco IOS Firewall for IPv6 enables you to implement Cisco IOS Firewall in IPv6 networks. Cisco IOS Firewall coexists with Cisco IOS Firewall for IPv4 networks and is supported on all dual-stack routers.

Cisco IOS Firewall for IPv6 features are as follows:

- Fragmented packet inspection--The fragment header is used to trigger fragment processing. Cisco IOS Firewall virtual fragment reassembly (VFR) examines out-of-sequence fragments and switches the packets into correct order, examines the number of fragments from a single IP given a unique identifier (Denial of Service [DoS] attack), and performs virtual reassembly to move packets to upper-layer protocols.
- IPv6 DoS attack mitigation--Mitigation mechanisms have been implemented in the same fashion as for IPv4 implementation, including SYN half-open connections.
- Tunneled packet inspection--Tunneled IPv6 packets terminated at a Cisco IOS firewall router can be inspected by the Cisco IOS Firewall for IPv6.
- Stateful packet inspection--The feature provides stateful packet inspection of TCP, UDP, Internet Control Message Protocol version 6 (ICMPv6), and FTP sessions.
- Stateful inspection of packets originating from the IPv4 network and terminating in an IPv6 environment--This feature uses IPv4-to-IPv6 translation services.
- Interpretation or recognition of most IPv6 extension header information--The feature provides IPv6 extension header information including routing header, hop-by-hop options header, and fragment header is interpreted or recognized.
- Port-to-application mapping (PAM)--Cisco IOS Firewall for IPv6 includes PAM.

- PAM in Cisco IOS Firewall for IPv6, page 795
- Cisco IOS Firewall Alerts Audit Trails and System Logging, page 796
- IPv6 Packet Inspection, page 796
- Cisco IOS Firewall Restrictions, page 796

PAM in Cisco IOS Firewall for IPv6

PAM allows you to customize TCP or UDP port numbers for network services or applications. PAM uses this information to support network environments that run services using ports that are different from the registered or well-known ports associated with an application.

Using the port information, PAM establishes a table of default port-to-application mapping information at the firewall. The information in the PAM table enables Context-based Access Control (CBAC) supported services to run on nonstandard ports. CBAC is limited to inspecting traffic using only the well-known or registered ports associated with an application, whereas PAM allows network administrators to customize network access control for specific applications and services.

PAM also supports host- or subnet-specific port mapping, which allows you to apply PAM to a single host or subnet using standard ACLs. Host- or subnet-specific port mapping is done using standard ACLs.

Cisco IOS Firewall Alerts Audit Trails and System Logging

Cisco IOS Firewall generates real-time alerts and audit trails based on events tracked by the firewall. Enhanced audit trail features use system logging to track all network transactions; to record time stamps, source host, destination host, and ports used; and to record the total number of transmitted bytes for advanced, session-based reporting. Real-time alerts send system logging error messages to central management consoles when the system detects suspicious activity. Using Cisco IOS Firewall inspection rules, you can configure alerts and audit trail information on a per-application protocol basis. For example, if you want to generate audit trail information for TCP traffic, you can specify the generation of this information in the Cisco IOS Firewall rule that defines TCP inspection.

The Cisco IOS Firewall provides audit trail messages to record details about inspected sessions. Audit trail information is configurable on a per-application basis using the CBAC inspection rules. To determine which protocol was inspected, use the port number associated with the responder. The port number appears immediately after the address.

IPv6 Packet Inspection

The following header fields are all used for IPv6 inspection--traffic class, flow label, payload length, next header, hop limit, and source or destination address. For further information on and descriptions of the IPv6 header fields, see RFC 2474.

Cisco IOS Firewall Restrictions

Cisco IOS Intrusion Detection System (IDS) is not supported for IPv6.

Zone-Based Policy Firewall IPv6 Support

The zone-based policy firewall for IPv6 coexists with the zone-based policy firewall for IPv4 in order to support IPv6 traffic. The feature provides MIB support for TCP, UDP, ICMPv6, and FTP sessions.

ACL--Hardware and Software Counters Granularity for IPv4 and IPv6 ACL Statistics

Each IPv6 and IPv4 ACL entry maintains a global counter per entry for the number of matches applied to the ACL entry. The counters reflect all matches applied to the ACL, regardless of where the match was applied (such as on the platform or in the software feature path). This feature allows both IPv4 and IPv6 ACLs on the Cisco Catalyst 6500 platform to update the ACL entry statistics with a platform entry count.

How to Implement Traffic Filters and Firewalls for IPv6 Security

Configuring IPv6 Traffic Filtering

If you are running Cisco IOS Release 12.2(13)T, 12.0(23)S, or later releases, proceed to the Creating and Configuring an IPv6 ACL for Traffic Filtering, page 797 section. If you are running Cisco IOS Release 12.2(11)T, 12.0(22)S, or earlier releases, proceed to the Creating an IPv6 ACL for Traffic Filtering in Cisco IOS Release 12.2(11)T 12.0(22)S or Earlier Releases, page 804 section.

Creating and Configuring an IPv6 ACL for Traffic Filtering

This section describes how to configure your networking devices to filter traffic, function as a firewall, or detect potential viruses.

Note

- Each IPv6 ACL contains implicit permit rules to enable IPv6 neighbor discovery. These rules can be overridden by the user by placing a deny ipv6 any any statement within an ACL. The IPv6 neighbor discovery process makes use of the IPv6 network layer service; therefore, by default, IPv6 ACLs implicitly allow IPv6 neighbor discovery packets to be sent and received on an interface. In IPv4, the Address Resolution Protocol (ARP), which is equivalent to the IPv6 neighbor discovery process, makes use of a separate data link layer protocol; therefore, by default, IPv4 ACLs implicitly allow ARP packets to be sent and received on an interface.
- Time-based and reflexive ACLs are not supported for IPv4 or IPv6 on the Cisco 12000 series platform. The **reflect**, **timeout**, and **time-range** keywords of the **permit** command in IPv6 are excluded on the Cisco 12000 series.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 access-list** *access-list-name*
4. Do one of the following:
   - **permit protocol** {*source-ipv6-prefix / prefix-length* | **any** | **host** *source-ipv6-address* | **auth**} [*operator* [*port-number*]] {*destination-ipv6-prefix / prefix-length*| **any** | **host** *destination-ipv6-address*| **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number*| *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**reflect** *name* [**timeout** *value*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*]
   -
   -
   - **deny** *protocol* {*source-ipv6-prefix / prefix-length* | **any** | **host** *source-ipv6-address* | **auth**} [*operator port-number*]] {*destination-ipv6-prefix/prefix-length* | **any** | **host** *destination-ipv6-address* | **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*] [**undetermined-transport**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 access-list** *access-list-name* **Example:** `Router(config)# ipv6 access-list outbound`	Defines an IPv6 ACL, and enters IPv6 access list configuration mode. • The *access-list name* argument specifies the name of the IPv6 ACL. IPv6 ACL names cannot contain a space or quotation mark, or begin with a numeral.

Command or Action	Purpose

Step 4 Do one of the following:
• **permit protocol** {*source-ipv6-prefix* / *prefix-length* | **any** | **host** *source-ipv6-address* | **auth**} [*operator* [*port-number*]] {*destination-ipv6-prefix* / *prefix-length*| **any** | **host** *destination-ipv6-address*| **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number*| *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**reflect** *name* [**timeout** *value*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*]
•
•
• **deny** *protocol* {*source-ipv6-prefix* / *prefix-length* | **any** | **host** *source-ipv6-address* | **auth**} [*operator port-number*]] {*destination-ipv6-prefix/ prefix-length* | **any** | **host** *destination-ipv6-address* | **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*] [**undetermined-transport**]
Example:
`Router(config-ipv6-acl)# permit tcp 2001:DB8:0300:0201::/32 eq telnet any reflect reflectout`
Example:
Example:
Example:
`Router(config-ipv6-acl)# deny tcp host 2001:DB8:1::1 any log-input`
	Specifies permit or deny conditions for an IPv6 ACL.

Applying the IPv6 ACL to an Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 traffic-filter** *access-list-name* {**in**| **out**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface ethernet 0`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**ipv6 traffic-filter** *access-list-name* {**in**| **out**} **Example:** `Router(config-if)# ipv6 traffic-filter outbound out`	Applies the specified IPv6 access list to the interface specified in the previous step.

Controlling Access to a vty

Creating an IPv6 ACL to Provide Access Class Filtering

Perform this task to control access to a vty on a router by creating an IPv6 ACL to provide access class filtering.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 access-list** *access-list-name*
4. Do one of the following:
   - **permit protocol** {*source-ipv6-prefix / prefix-length* | **any** | **host** *source-ipv6-address* | **auth**} [*operator* [*port-number*]] {*destination-ipv6-prefix / prefix-length*| **any** | **host** *destination-ipv6-address*| **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number*| *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**reflect** *name* [**timeout** *value*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*]
   -
   -
   - **deny** *protocol* {*source-ipv6-prefix / prefix-length* | **any**| **host** *source-ipv6-address* | **auth**} [*operator*[*port-number*]] {*destination-ipv6-prefix/prefix-length* | **any**| **host** *destination-ipv6-address* | **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*] [**undetermined-transport**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 access-list** *access-list-name* **Example:** `Router(config)# ipv6 access-list cisco`	Defines an IPv6 ACL, and enters IPv6 access list configuration mode.

Command or Action	Purpose

Step 4 Do one of the following:
• **permit protocol** {*source-ipv6-prefix* / *prefix-length* | **any** | **host** *source-ipv6-address* | **auth**} [*operator* [*port-number*]] {*destination-ipv6-prefix* / *prefix-length*| **any** | **host** *destination-ipv6-address*| **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number*| *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**reflect** *name* [**timeout** *value*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*]
•
•
• **deny** *protocol* {*source-ipv6-prefix* / *prefix-length* | **any**| **host** *source-ipv6-address* | **auth**} [*operator*[*port-number*]] {*destination-ipv6-prefix/prefix-length* | **any**| **host** *destination-ipv6-address* | **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*] [**undetermined-transport**]
Example:
`Router(config-ipv6-acl)# permit ipv6 host 2001:DB8:0:4::32 any eq telnet`
Example:
Example:
`Router(config-ipv6-acl)# deny ipv6 host 2001:DB8:0:6::6/32 any`
	Specifies permit or deny conditions for an IPv6 ACL.

Applying an IPv6 ACL to the Virtual Terminal Line

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **line** [**aux**| **console**| **tty**| **vty**] *line-number*[*ending-line-number*]
4. **ipv6 access-class** *ipv6-access-list-name* {**in**| **out**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**line** [**aux**| **console**| **tty**| **vty**] *line-number*[*ending-line-number*] **Example:** `Router(config)# line vty 0 4`	Identifies a specific line for configuration and enters line configuration mode. • In this example, the **vty** keyword is used to specify the virtual terminal lines for remote console access.
Step 4	**ipv6 access-class** *ipv6-access-list-name* {**in**| **out**} **Example:** `Router(config-line)# ipv6 access-class cisco in`	Filters incoming and outgoing connections to and from the router based on an IPv6 ACL.

Configuring TCP or UDP Matching

TCP or UDP traffic can be matched to the ULP (for example, TCP, UDP, ICMP, SCTP) if an AH is present or absent. Before this feature was introduced, this function was only available if an AH was absent.

Use of the keyword **auth** with the **permit icmp** and **deny icmp** commands allows TCP or UDP traffic to be matched to the ULP if an AH is present. TCP or UDP traffic without an AH will not be matched.

IPv6 traffic can be matched to a ULP when an AH header is present. To perform this function, enter the **ahp** option for the *protocol* argument when using the **permit** or **deny** command.

Perform this task to allow TCP or UDP traffic to be matched to the ULP if an AH is present.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 access-list** *access-list-name*
4. **permit icmp auth**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 access-list** *access-list-name* **Example:** `Router(config)# ipv6 access-list list1`	Defines an IPv6 access list and places the router in IPv6 access list configuration mode.

Step 4	**permit icmp auth**
	
	Example:
	
	Example:
	`or`
	
	Example:
	deny icmp auth
	
	Example:
	`Router(config-ipv6-acl)# permit icmp auth`
		Specifies permit or deny conditions for an IPv6 ACL using the **auth** keyword, which is used to match against the presence of the AH.

Creating an IPv6 ACL for Traffic Filtering in Cisco IOS Release 12.2(11)T 12.0(22)S or Earlier Releases

Perform the following tasks to create and apply ACLs in Cisco IOS Release 12.2(11)T, 12.0(22)S, or earlier releases.

Creating an IPv6 ACL in Cisco IOS Release 12.2(11)T 12.0(22)S or Earlier Releases

Perform this task to create an IPv6 ACL and configure the IPv6 ACL to pass or block traffic in Cisco IOS Release 12.2(11)T, 12.0(22)S, or earlier releases.

Note

- The *source-ipv6-prefix* argument filters traffic by packet source address, and the *destination-ipv6-prefix* argument filters traffic by packet destination address.
- The Cisco IOS software compares an IPv6 prefix against the permit and deny condition statements in the access list. Every IPv6 access list, including access lists that do not have any permit and deny condition statements, has an implicit deny any any statement as its last match condition. The priority or sequence value applied to each condition statement dictates the order in which the statement is applied in the access list.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 access-list** *access-list-name* {**permit** | **deny**} {*source-ipv6-prefix / prefix-length*| **any**} {*destination-ipv6-prefix / prefix-length*| **any**} [**priority** *value*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

Command or Action	Purpose
Step 3 **ipv6 access-list** *access-list-name* {**permit** | **deny**} {*source-ipv6-prefix* / *prefix-length*| **any**} {*destination-ipv6-prefix* / *prefix-length*| **any**} [**priority** *value*] **Example:** `Router(config)# ipv6 access-list list2 deny fec0:0:0:2::/64 any`	Creates an IPv6 ACL and sets deny or permit conditions for the ACL.

Applying the IPv6 ACL to an Interface in Cisco IOS Release 12.2(11)T 12.0(22)S or Earlier Releases

Perform this task to apply the IPv6 ACL to an interface in Cisco IOS Release 12.2(11)T, 12.0(22)S, or earlier releases.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 traffic-filter** *access-list-name* {**in**\| **out**}

DETAILED STEPS

Command or Action	Purpose
Step 1 **enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2 **configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3 **interface** *type number* **Example:** `Router(config)# interface ethernet 0`	Specifies the interface type and number, and enters interface configuration mode.

	Command or Action	Purpose
Step 4	**ipv6 traffic-filter** *access-list-name* {**in**| **out**} **Example:** `Router(config-if)# ipv6 traffic-filter list2 out`	Applies the specified IPv6 access list to the interface specified in the previous step.

Configuring the Cisco IOS Firewall for IPv6

This configuration scenario uses both packet inspection and ACLs.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **ipv6 inspect name** *inspection-name protocol* [**alert** {**on** | **off**}] [**audit-trail**{**on** | **off**}] [**timeout** *seconds*]
5. **interface** *type number*
6. **ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits* / *prefix-length*
7. **ipv6 enable**
8. **ipv6 traffic-filter** *access-list-name* {**in** | **out**
9. **ipv6 inspect** *inspection-name* {**in** | **out**
10. **ipv6 access-list** *access-list-name*
11. Do one of the following:
    - **permit protocol** {*source-ipv6-prefix/prefix-length* | **any** | **host** *source-ipv6-address* | **auth**} [*operator* [*port-number*]] {*destination-ipv6-prefix* / *prefix-length*| **any** | **host** *destination-ipv6-address*| **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number*| *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**reflect** *name* [**timeout** *value*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*]
    -
    -
    - **deny** *protocol* {*source-ipv6-prefix* / *prefix-length* | **any**| **host** *source-ipv6-address* | **auth**} [*operator*[*port-number*]] {*destination-ipv6-prefix/prefix-length* | **any host** *destination-ipv6-address* | **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name* **]** [**undetermined-transport**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Router(config)# ipv6 unicast-routing`	Enables IPv6 unicast routing.
Step 4	**ipv6 inspect name** *inspection-name protocol* [**alert** {**on** | **off**}] [**audit-trail**{**on** | **off**}] [**timeout** *seconds*] **Example:** `Router(config)# ipv6 inspect name ipv6_test icmp timeout 60`	Defines a set of IPv6 inspection rules for the firewall.
Step 5	**interface** *type number* **Example:** `Router(config)# interface FastEthernet0/0`	Specifies the interface on which the inspection will occur.
Step 6	**ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits* / *prefix-length* **Example:** `Router(config-if)# ipv6 address 3FFE:C000:0:7::/64 eui-64`	Provides the address for the inspection interface.
Step 7	**ipv6 enable** **Example:** `Router(config-if)# ipv6 enable`	Enables IPv6 routing. **Note** This step is optional if the IPv6 address is specified in step 6.

	Command or Action	Purpose
Step 8	**ipv6 traffic-filter** *access-list-name* {**in** | **out** **Example:** `Router(config-if)# ipv6 traffic-filter outbound out`	Applies the specified IPv6 access list to the interface specified in the previous step.
Step 9	**ipv6 inspect** *inspection-name* {**in** | **out** **Example:** `Router(config)# ipv6 inspect ipv6_test in`	Applies the set of inspection rules.
Step 10	**ipv6 access-list** *access-list-name* **Example:** `Router(config)# ipv6 access-list outbound`	Defines an IPv6 ACL and enters IPv6 access list configuration mode. The router prompt changes to Router(config-ipv6-acl)#.

Command or Action	Purpose

Step 11 Do one of the following:
• **permit protocol** {*source-ipv6-prefix/prefix-length* | **any** | **host** *source-ipv6-address* | **auth**} [*operator* [*port-number*]] {*destination-ipv6-prefix / prefix-length*| **any** | **host** *destination-ipv6-address*| **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number*| *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**reflect** *name* [**timeout** *value*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*]
•
•
• **deny** *protocol* {*source-ipv6-prefix / prefix-length* | **any**| **host** *source-ipv6-address* | **auth**} [*operator*[*port-number*]] {*destination-ipv6-prefix/prefix-length* | **any host** *destination-ipv6-address* | **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name* **]** [**undetermined-transport**]
Example:
`Router(config-ipv6-acl)# permit tcp 2001:DB8:0300:0201::/32 any reflect reflectout`
Example:
Example:
`Router(config-ipv6-acl)# deny tcp fec0:0:0:0201::/64 any`
	Specifies permit or deny conditions for an IPv6 ACL.

• Configuring PAM for IPv6, page 810

Configuring PAM for IPv6

Creating an IPv6 Access Class Filter for PAM

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 access-list** *access-list-name*
4. Do one of the following:
   - **permit protocol** {*source-ipv6-prefix/prefix-length* | **any** | **host***source-ipv6-address* | **auth**} [*operator* [*port-number*]] {*destination-ipv6-prefix /prefix-length* | **any** | **host***destination-ipv6-address* | **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp***value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**reflect** *name* [**timeout** *value*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*]
   -
   -
   - **deny** *protocol source-ipv6-prefix / prefix-length* | **any** | **host** *source-ipv6-address* | **auth**} [*operator port-number*]] *destination-ipv6-prefix/prefix-length* **any host** *destination-ipv6-address* | **auth**} [*operator port-number*]] **dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value* **flow-label** *value* **fragments log log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name* **undetermined-transport**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 access-list** *access-list-name* **Example:** `Router(config)# ipv6 access-list outbound`	Defines an IPv6 ACL and enters IPv6 access list configuration mode.

Command or Action	Purpose

Step 4 Do one of the following:
• **permit protocol** {*source-ipv6-prefix/prefix-length* | **any** | **host***source-ipv6-address* | **auth**} [*operator* [*port-number*]] {*destination-ipv6-prefix /prefix-length* | **any** | **host***destination-ipv6-address* | **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp***value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**reflect** *name* [**timeout** *value*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*]
•
•
• **deny** *protocol source-ipv6-prefix / prefix-length* | **any** | **host** *source-ipv6-address* | **auth**} [*operator port-number*]] *destination-ipv6-prefix/prefix-length* **any host** *destination-ipv6-address* | **auth**} [*operator port-number*]] **dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value* **flow-label** *value* **fragments log log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name* **undetermined-transport**
Example:
`Router(config-ipv6-acl)# permit tcp 2001:DB8:0300:0201::/32 any reflect reflectout`
Example:
Example:
`Router(config-ipv6-acl)# deny tcp fec0:0:0:0201::/64 any`
	Specifies permit or deny conditions for an IPv6 ACL.

Applying the IPv6 Access Class Filter to PAM

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 port-map** *application-name* **port** *port-num* [**list** *acl-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 port-map** *application-name* **port** *port-num* [**list** *acl-name* **Example:** `Router(config)# ipv6 port-map ftp port 8090 list PAMACL`	Establishes PAM for the system.

Configuring Zone-Based Firewall in IPv6

Configuring an Inspect-Type Parameter Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **parameter-map type inspect** {*parameter-map-name* | **global** | **default**}
4. **sessions maximum** *sessions*
5. **ipv6 routing-enforcement-header loose**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**parameter-map type inspect** {*parameter-map-name* | **global** | **default**} **Example:** `Router(config)# parameter-map type inspect v6-param-map`	Configures an inspect type parameter map for connecting thresholds, timeouts, and other parameters pertaining to the inspect action, and places the router in parameter map configuration mode.
Step 4	**sessions maximum** *sessions* **Example:** `Router(config-profile)# sessions maximum 10000`	Sets the maximum number of allowed sessions that can exist on a zone pair.
Step 5	**ipv6 routing-enforcement-header loose** **Example:** `Router(config-profile)# ipv6 routing-enforcement-header loose`	Provides backward compatibility with legacy IPv6 inspection.

Creating and Using an Inspect-Type Class Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map type inspect** {**match-any** | **match-all**} *class-map-name*
4. **match protocol tcp**
5. **match protocol udp**
6. **match protocol icmp**
7. **match protocol ftp**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**class-map type inspect** {**match-any** | **match-all**} *class-map-name* **Example:** `Router(config-profile)# class-map type inspect match-any v6-class`	Create an inspect type class map, and places the router in lass-map configuration mode.
Step 4	**match protocol tcp** **Example:** `Router(config-cmap)# match protocol tcp`	Configures the match criterion for a class map based on TCP.
Step 5	**match protocol udp** **Example:** `Router(config-cmap)# match protocol udp`	Configures the match criterion for a class map based on UDP.

	Command or Action	Purpose
Step 6	**match protocol icmp** **Example:** `Router(config-cmap)# match protocol icmp`	Configures the match criterion for a class map based on ICMP.
Step 7	**match protocol ftp** **Example:** `Router(config-cmap)# match protocol ftp`	Configures the match criterion for a class map based on FTP.

Creating and Using an Inspect-Type Policy Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map type inspect** *policy-map-name*
4. **class type inspect** *class-map-name*
5. **inspect** [*parameter-map-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map type inspect** *policy-map-name* **Example:** `Router(config)# policy-map type inspect v6-policy`	Creates an inspect-type policy map, and places the router in policy-map configuration mode.

	Command or Action	Purpose
Step 4	**class type inspect** *class-map-name* **Example:** `Router(config-pmap)# class type inspect v6-class`	Specifies the traffic (class) on which an action is to be performed.
Step 5	**inspect** [*parameter-map-name*] **Example:** `Router(config-pmap)# inspect`	Enables Cisco IOS stateful packet inspection.

Creating Security Zones and Zone Pairs

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **zone security** {*zone-name* | **default**}
4. **zone security** {*zone-name* | **default**}
5. **zone-pair security** *zone-pair-name* **source** {*source-zone-name* | **self** | **default**} **destination** {*destination-zone-name* | **self** | **default**}
6. **service-policy type inspect** *policy-map-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**zone security** {*zone-name* | **default**} **Example:** `Router(config)# zone security 1`	Creates a security zone. • Cisco recommends that you create at least two security zones so that you can create a zone pair.
Step 4	**zone security** {*zone-name* | **default**} **Example:** `Router(config)# zone security 2`	Creates a security zone. • Cisco recommends that you create at least two security zones so that you can create a zone pair.
Step 5	**zone-pair security** *zone-pair-name* **source** {*source-zone-name* | **self** | **default**} **destination** {*destination-zone-name* | **self** | **default**} **Example:** `Router(config)# zone-pair security zp source z1 destination z2`	Creates a zone pair, and places the router in zone-pair configuration mode.
Step 6	**service-policy type inspect** *policy-map-name* **Example:** `Router(config-sec-zone-pair)# service-policy type inspect v6-policy`	Attaches a firewall policy map to a zone pair.

Configuring ACL Hardware and Software Counters Granularity for IPv4 and IPv6 ACL Statistics

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 access-list** *access-list-name*
4. **hardware statistics**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 access-list** *access-list-name* **Example:** `Router(config)# ipv6 access-list outbound`	Defines an IPv6 ACL and enters IPv6 access list configuration mode.
Step 4	**hardware statistics** **Example:** `Router(config-ipv6-acl)# hardware statistics`	Enables the collection of hardware statistics.

Verifying IPv6 Security Configuration and Operation

SUMMARY STEPS

1. **show crypto ipsec sa** [**map** *map-name* | **address** | **identity** | **interface** *interface-type interface-number* | **peer** [**vrf** *fvrf-name*] **address** | **vrf** *ivrf-name* | **ipv6** [*interface-type interface-number*]] [**detail**
2. **show crypto isakmp peer** [**config** | **detail**]
3. **show crypto isakmp profile**
4. **show crypto isakmp sa** [**active** | **standby** | **detail** | **nat**
5. **show ipv6 access-list** [*access-list-name*]
6. **show ipv6 inspect** {**name** *inspection-name* | **config** | **interfaces** | **session** [**detail**] | **all**}
7. **show ipv6 port-map** [*application* | **port** *port-number*
8. **show ipv6 prefix-list** [**detail** | **summary**] [*list-name*
9. **show ipv6 virtual-reassembly interface** *interface-type*
10. **show logging** [**slot** *slot-number* | **summary**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show crypto ipsec sa** [**map** *map-name* | **address** | **identity** | **interface** *interface-type interface-number* | **peer** [**vrf** *fvrf-name*] **address** | **vrf** *ivrf-name* | **ipv6** [*interface-type interface-number*]] [**detail** **Example:** `Router# show crypto ipsec sa ipv6`	Displays the settings used by current SAs.
Step 2	**show crypto isakmp peer** [**config** | **detail**] **Example:** `Router# show crypto isakmp peer`	Displays peer descriptions.
Step 3	**show crypto isakmp profile** **Example:** `Router# show crypto isakmp profile`	Lists all the ISAKMP profiles that are defined on a router.
Step 4	**show crypto isakmp sa** [**active** | **standby** | **detail** | **nat** **Example:** `Router# show crypto isakmp sa`	Displays current IKE SAs.
Step 5	**show ipv6 access-list** [*access-list-name*] **Example:** `Router# show ipv6 access-list`	Displays the contents of all current IPv6 access lists.
Step 6	**show ipv6 inspect** {**name** *inspection-name* | **config** | **interfaces** | **session** [**detail**] | **all**} **Example:** `Router# show ipv6 inspect interfaces`	Displays CBAC configuration and session information.

	Command or Action	Purpose
Step 7	**show ipv6 port-map** [*application* | **port** *port-number* **Example:** `Router# show ipv6 port-map ftp`	Displays PAM configuration.
Step 8	**show ipv6 prefix-list** [**detail** | **summary**] [*list-name* **Example:** `Router# show ipv6 prefix-list`	Displays information about an IPv6 prefix list or IPv6 prefix list entries.
Step 9	**show ipv6 virtual-reassembly interface** *interface-type* **Example:** `Router# show ipv6 virtual-reassembly interface e1/1`	Displays configuration and statistical information of VFR.
Step 10	**show logging** [**slot** *slot-number* | **summary** **Example:** `Router# show logging`	Displays the state of system logging (syslog) and the contents of the standard system logging buffer. • Access list entries with the **log** or **log-input**keywords will be logged when a packet matches the access list entry.

Troubleshooting IPv6 Security Configuration and Operation

SUMMARY STEPS

1. **enable**
2. **clear ipv6 access-list** [*access-list-name*]
3. **clea r ipv6 inspect** {**session** *session-number* | all}
4. **clear ipv6 prefix-list** [*prefix-list-name*] [*ipv6-prefix* / *prefix-length*
5. **debug crypto ipsec**
6. **debug crypto engine packet** [**detail**
7. **debug ipv6 inspect** {**function-trace** | **object-creation** | **object-deletion** | **events** | **timers** | **protocol** | **detailed**
8. **debug ipv6 packet** [**access-list** *access-list-name*] [**detail**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ipv6 access-list** [*access-list-name*] **Example:** `Router# clear ipv6 access-list tin`	Resets the IPv6 access list match counters.
Step 3	**clea r ipv6 inspect** {**session** *session-number* | all} **Example:** `Router# clear ipv6 inspect all`	Removes a specific IPv6 session or all IPv6 inspection sessions.
Step 4	**clear ipv6 prefix-list** [*prefix-list-name*] [*ipv6-prefix* / *prefix-length* **Example:** `Router# clear ipv6 prefix-list`	Resets the hit count of the IPv6 prefix list entries.
Step 5	**debug crypto ipsec** **Example:** `Router# debug crypto ipsec`	Displays IPsec network events.
Step 6	**debug crypto engine packet** [**detail** **Example:** `Router# debug crypto engine packet`	Displays the contents of IPv6 packets. **Caution** Using this command could flood the system and increase CPU usage if several packets are being encrypted.
Step 7	**debug ipv6 inspect** {**function-trace** | **object-creation** | **object-deletion** | **events** | **timers** | **protocol** | **detailed** **Example:** `Router# debug ipv6 inspect timers`	Displays messages about Cisco IOS Firewall events.

Command or Action	Purpose
Step 8 **debug ipv6 packet** [**access-list** *access-list-name*] [**detail**] **Example:** `Router# debug ipv6 packet access-list PAK-ACL`	Displays debugging messages for IPv6 packets.

- Examples, page 823

Examples

Sample Output from the show crypto ipsec sa ipv6 Command

The following is sample output from the **show crypto ipsec sa ipv6** command:

```
Router# show crypto ipsec sa ipv6
interface: Tunnel0
     Crypto map tag: Tunnel0-head-0, local addr 3FFE:2002::A8BB:CCFF:FE01:9002
    protected vrf: (none)
    local  ident (addr/mask/prot/port): (::/0/0/0)
    remote ident (addr/mask/prot/port): (::/0/0/0)
    current_peer 3FFE:2002::A8BB:CCFF:FE01:2C02 port 500
      PERMIT, flags={origin_is_acl,}
     #pkts encaps: 133, #pkts encrypt: 133, #pkts digest: 133
     #pkts decaps: 133, #pkts decrypt: 133, #pkts verify: 133
     #pkts compressed: 0, #pkts decompressed: 0
     #pkts not compressed: 0, #pkts compr. failed: 0
     #pkts not decompressed: 0, #pkts decompress failed: 0
     #send errors 60, #recv errors 0
      local crypto endpt.: 3FFE:2002::A8BB:CCFF:FE01:9002,
      remote crypto endpt.: 3FFE:2002::A8BB:CCFF:FE01:2C02
      path mtu 1514, ip mtu 1514
      current outbound spi: 0x28551D9A(676666778)
      inbound esp sas:
       spi: 0x2104850C(553944332)
         transform: esp-des ,
         in use settings ={Tunnel, }
         conn id: 93, flow_id: SW:93, crypto map: Tunnel0-head-0
         sa timing: remaining key lifetime (k/sec): (4397507/148)
         IV size: 8 bytes
         replay detection support: Y
         Status: ACTIVE
      inbound ah sas:
       spi: 0x967698CB(2524354763)
         transform: ah-sha-hmac ,
         in use settings ={Tunnel, }
         conn id: 93, flow_id: SW:93, crypto map: Tunnel0-head-0
         sa timing: remaining key lifetime (k/sec): (4397507/147)
         replay detection support: Y
         Status: ACTIVE
      inbound pcp sas:
      outbound esp sas:
       spi: 0x28551D9A(676666778)
         transform: esp-des ,
         in use settings ={Tunnel, }
         conn id: 94, flow_id: SW:94, crypto map: Tunnel0-head-0
         sa timing: remaining key lifetime (k/sec): (4397508/147)
         IV size: 8 bytes
         replay detection support: Y
         Status: ACTIVE
      outbound ah sas:
       spi: 0xA83E05B5(2822636981)
```

```
        transform: ah-sha-hmac ,
        in use settings ={Tunnel, }
        conn id: 94, flow_id: SW:94, crypto map: Tunnel0-head-0
        sa timing: remaining key lifetime (k/sec): (4397508/147)
        replay detection support: Y
        Status: ACTIVE
      outbound pcp sas:
```

Sample Output from the show crypto isakmp peer Command

The following sample output shows peer descriptions on an IPv6 router:

```
Router# show crypto isakmp peer detail
Peer: 2001:DB8:0:1::1 Port: 500 Local: 2001:DB8:0:2::1
Phase1 id: 2001:DB8:0:1::1
flags:
NAS Port: 0 (Normal)
IKE SAs: 1 IPsec SA bundles: 1
last_locker: 0x141A188, last_last_locker: 0x0
last_unlocker: 0x0, last_last_unlocker: 0x0
```

Sample Output from the show crypto isakmp profile Command

The following sample output shows the ISAKMP profiles that are defined on an IPv6 router:

```
Router# show crypto isakmp profile
ISAKMP PROFILE tom
Identities matched are:
ipv6-address 2001:DB8:0:1::1/32
Certificate maps matched are:
Identity presented is: ipv6-address fqdn
keyring(s): <none>
trustpoint(s): <all>
```

Sample Output from the show crypto isakmp sa Command

The following sample output shows the SAs of an active IPv6 device. The IPv4 device is inactive:

```
Router# show crypto isakmp sa detail

Codes: C - IKE configuration mode, D - Dead Peer Detection
       K - Keepalives, N - NAT-traversal
       X - IKE Extended Authentication
       psk - Preshared key, rsig - RSA signature
       renc - RSA encryption
IPv4 Crypto ISAKMP SA

C-id  Local           Remote          I-VRF    Status Encr Hash Auth DH
Lifetime Cap.

IPv6 Crypto ISAKMP SA

  dst: 3FFE:2002::A8BB:CCFF:FE01:2C02
  src: 3FFE:2002::A8BB:CCFF:FE01:9002
  conn-id: 1001  I-VRF:          Status: ACTIVE Encr: des  Hash: sha  Auth:
psk
  DH: 1  Lifetime: 23:45:00 Cap: D    Engine-id:Conn-id = SW:1

  dst: 3FFE:2002::A8BB:CCFF:FE01:2C02
  src: 3FFE:2002::A8BB:CCFF:FE01:9002
  conn-id: 1002  I-VRF:          Status: ACTIVE Encr: des  Hash: sha  Auth:
psk
  DH: 1  Lifetime: 23:45:01 Cap: D    Engine-id:Conn-id = SW:2
```

Sample Output from the show ipv6 access-list Command

In the following example, the **show ipv6 access-list**command is used to verify that IPv6 ACLs are configured correctly:

```
Router> show ipv6 access-list
IPv6 access list inbound
```

```
    permit tcp any any eq bgp reflect tcptraffic (8 matches) sequence 10
    permit tcp any any eq telnet reflect tcptraffic (15 matches) sequence 20
    permit udp any any reflect udptraffic sequence 30
IPv6 access list tcptraffic (reflexive) (per-user)
    permit tcp host 2001:DB8:1::32 eq bgp host 2001:DB8:2::32 eq 11000 timeout 300
(time         left 243) sequence 1
    permit tcp host 2001:DB8:1::32 eq telnet host 2001:DB8:2::32 eq 11001 timeout
300         (time left 296) sequence 2
IPv6 access list outbound
    evaluate udptraffic
    evaluate tcptraffic
```

Sample Output from the show ipv6 prefix-list Command

The following example shows the output of the **show ipv6 prefix-list** command with the **detail** keyword:

```
Router# show ipv6 prefix-list detail
Prefix-list with the last deletion/insertion: bgp-in
ipv6 prefix-list 6to4:
   count: 1, range entries: 0, sequences: 5 - 5, refcount: 2
   seq 5 permit 2001:DB8::/32 (hit count: 313, refcount: 1)
ipv6 prefix-list aggregate:
   count: 2, range entries: 2, sequences: 5 - 10, refcount: 30
   seq 5 deny 3FFE:C00::/24 ge 25 (hit count: 568, refcount: 1)
   seq 10 permit ::/0 le 48 (hit count: 31310, refcount: 1)
ipv6 prefix-list bgp-in:
   count: 6, range entries: 3, sequences: 5 - 30, refcount: 31
   seq 5 deny 5F00::/8 le 128 (hit count: 0, refcount: 1)
   seq 10 deny ::/0 (hit count: 0, refcount: 1)
   seq 15 deny ::/1 (hit count: 0, refcount: 1)
   seq 20 deny ::/2 (hit count: 0, refcount: 1)
   seq 25 deny ::/3 ge 4 (hit count: 0, refcount: 1)
   seq 30 permit ::/0 le 128 (hit count: 240664, refcount: 0)
```

Sample Output from the show ipv6 virtual-reassembly Command

The following example shows the output of the **show ipv6 virtual-reassembly**command with the **interface**keyword:

```
Router# show ipv6 virtual-reassembly interface e1/1
Configuration Information:
--------------------------------
Virtual Fragment Reassembly (VFR) is ENABLED...
Maximum number of datagram that can be reassembled at a time: 64
Maximum number of fragments per datagram: 8
Timeout value of a datagram: 3 seconds
Statistical Information:
---------------------------
Number of datagram being reassembled:12
Number of fragments being processed:48
Total number of datagram reassembled:6950
Total number of datagram failed: 9
```

Sample Output from the show logging Command

In the following example, the **show logging**command is used to display logging entries that match the first line (sequence 10) of the access list named list1:

```
Router> show logging
00:00:36: %IPV6-6-ACCESSLOGP: list list1/10 permitted tcp 2001:DB8:1::1(11001)
(Ethernet0/0) -> 2001:DB8:1::2(179), 1 packet
```

Sample Output from the clear ipv6 access-list Command

In the following example, the **show ipv6 access-list**command is used to display some match counters for the access list named list1. The **clear ipv6 access-list**command is issued to reset the match counters for the

access list named list1. The **show ipv6 access-list**command is used again to show that the match counters have been reset.

```
Router> show ipv6 access-list list1
IPv6 access list list1
    permit tcp any any log-input (6 matches) sequence 10
    permit icmp any any echo-request log-input sequence 20
    permit icmp any any echo-reply log-input sequence 30
Router# clear ipv6 access-list list1
Router# show ipv6 access-list list1
IPv6 access list list1
    permit tcp any any log-input sequence 10
    permit icmp any any echo-request log-input sequence 20
    permit icmp any any echo-reply log-input sequence 30
```

Configuration Examples for Implementing Traffic Filters and Firewalls for IPv6 Security

Examples Creating and Applying IPv6 ACLs

Example: Creating and Applying an IPv6 ACL

This example configures two IPv6 ACLs named OUTBOUND and INBOUND and applies both ACLs to outbound and inbound traffic on Ethernet interface 0. The first and second permit entries in the OUTBOUND list permit all TCP and User Datagram Protocol (UDP) packets from network 2001:DB8:0300:0201::/32 to exit out of Ethernet interface 0. The entries also configure the temporary IPv6 reflexive ACL named REFLECTOUT to filter returning (incoming) TCP and UDP packets on Ethernet interface 0. The first deny entry in the OUTBOUND list keeps all packets from the network fec0:0:0:0201::/64 (packets that have the site-local prefix fec0:0:0:0201 as the first 64 bits of their source IPv6 address) from exiting out of Ethernet interface 0.

The **evaluate** command in the INBOUND list applies the temporary IPv6 reflexive ACL named REFLECTOUT to inbound TCP and UDP packets on Ethernet interface 0. When outgoing TCP or UDP packets are permitted on Ethernet interface 0 by the OUTBOUND list, the INBOUND list uses the REFLECTOUT list to match (evaluate) the returning (incoming) TCP and UDP packets.

```
ipv6 access-list OUTBOUND
 permit tcp 2001:DB8:0300:0201::/32 any reflect REFLECTOUT
 permit udp 2001:DB8:0300:0201::/32 any reflect REFLECTOUT
 deny fec0:0:0:0201::/64 any
ipv6 access-list INBOUND
 evaluate REFLECTOUT
```

```
interface ethernet 0
 ipv6 traffic-filter OUTBOUND out
 ipv6 traffic-filter INBOUND in
```

Note Given that a **permit any any** statement is not included as the last entry in the OUTBOUND or INBOUND ACL, only TCP and UDP packets matching the configured permit entries in the ACL and ICMP packets matching the implicit permit conditions in the ACL are permitted out of and in to Ethernet interface 0 (the implicit deny all condition at the end of the ACL denies all other packet types on the interface).

The example configures HTTP access to be restricted to certain hours during the day, and to log any activity outside of the permitted hours:

```
time-range lunchtime
 periodic weekdays 12:00 to 13:00
ipv6 access-list OUTBOUND
 permit tcp any any eq www time-range lunchtime
 deny tcp any any eq www log-input
 permit tcp 2001:DB8::/32 any
 permit udp 2001:DB8::/32 any
```

Example Creating and Applying an IPv6 ACL for 12.2(11)T 12.0(22)S or Earlier Releases

The following example is from a router running Cisco IOS Release 12.2(11)T, 12.0(22)S, or earlier releases.

The example configures the IPv6 ACL named list2 and applies the ACL to outbound traffic on Ethernet interface 0. Specifically, the first ACL entry keeps all packets from the network fec0:0:0:2::/64 (packets that have the site-local prefix fec0:0:0:2 as the first 64 bits of their source IPv6 address) from exiting out of Ethernet interface 0. The second entry in the ACL permits all other traffic to exit out of Ethernet interface 0. The second entry is necessary because an implicit deny all condition is at the end of each IPv6 ACL.

```
ipv6 access-list list2 deny fec0:0:0:2::/64 any
ipv6 access-list list2 permit any any
interface ethernet 0
 ipv6 traffic-filter list2 out
```

If the same configuration was used on a router running Cisco IOS Release 12.2(13)T, 12.0(23)S, or later releases, the configuration would be translated into IPv6 access list configuration mode as follows:

```
ipv6 access-list list2
 deny ipv6 fec0:0:0:2::/64 any
 permit ipv6 any any
interface ethernet 0
 ipv6 traffic-filter list2 out
```

Note IPv6 is automatically configured as the protocol type in **permit any any** and **deny any any** statements that are translated from global configuration mode to IPv6 access list configuration mode.

Example Controlling Access to a vty

In the following example, incoming connections to the virtual terminal lines 0 to 4 are filtered based on the IPv6 access list named acl1:

```
ipv6 access-list acl1
 permit ipv6 host 2001:DB8:0:4::2/32 any
```

```
!
line vty 0 4
 ipv6 access-class acl1 in
```

Example: Configuring TCP or UDP Matching

The following example allows any TCP traffic regardless of whether or not an AH is present:

```
IPv6 access list example1
    permit tcp any any
```

The following example allows TCP or UDP parsing only when an AH header is present. TCP or UDP traffic without an AH will not be matched:

```
IPv6 access list example2
    deny tcp host 2001::1 any log sequence 5
    permit tcp any any auth sequence 10
    permit udp any any auth sequence 20
```

The following example allows any IPv6 traffic containing an authentication header:

```
IPv6 access list example3
    permit ahp any any
```

Example: Configuring Cisco IOS Firewall for IPv6

This Cisco IOS Firewall configuration example uses inbound and outbound filters for inspection and makes use of access lists to manage the traffic. The inspect mechanism is the method of permitting return traffic based upon a packet being valid for an existing session for which the state is being maintained:

```
enable
configure terminal
 ipv6 unicast-routing
  ipv6 inspect name ipv6_test icmp timeout 60
  ipv6 inspect name ipv6_test tcp timeout 60
  ipv6 inspect name ipv6_test udp timeout 60

interface FastEthernet0/0
  ipv6 address 3FFE:C000:0:7::/64 eui-64
  ipv6 enable
  ipv6 traffic-filter INBOUND out
  ipv6 inspect ipv6_test in

interface FastEthernet0/1
  ipv6 address 3FFE:C000:1:7::/64 eui-64
  ipv6 enable
  ipv6 traffic-filter OUTBOUND in

! This is used for 3745b connection to tftpboot server
interface FastEthernet4/0
  ip address 192.168.17.33 255.255.255.0
  duplex auto
  speed 100

ip default-gateway 192.168.17.8
! end of tftpboot server config

! Access-lists to deny everything except for Neighbor Discovery ICMP messages
ipv6 access-list INBOUND
  permit icmp any any nd-na
  permit icmp any any nd-ns
  deny ipv6 any any log

ipv6 access-list OUTBOUND
  permit icmp any any nd-na
```

```
  permit icmp any any nd-ns
  deny ipv6 any any log
```

Example: Configuring Cisco IOS Zone-Based Firewall for IPv6

```
parameter-map type inspect v6-param-map
 sessions maximum 10000
 ipv6 routing-header-enforcement loose
!
!
class-map type inspect match-any v6-class
 match protocol tcp
 match protocol udp
 match protocol icmp
 match protocol ftp
!
!
policy-map type inspect v6-policy
 class type inspect v6-class
  inspect
!
zone security z1
zone security z2
!
zone-pair security zp source z1 destination z2
 service-policy type inspect v6-policy
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 IPsec	" Implementing IPsec in IPv6 Security ," *Cisco IOS IPv6 Configuration Guide*
Basic IPv6 configuration	" Implementing IPv6 Addressing and Basic Connectivity ," *Cisco IOS IPv6 Configuration Guide*
Zone-based firewalls	" Zone-Based Policy Firewall ," *Cisco IOS Security Configuration Guide: Securing the Data Plane*
IPv6 supported feature list	"Start Here: Cisco IOS Software Release Specifics for IPv6 Features ," *Cisco IOS IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIBs	MIBs Link
• CISCO-UNIFIED-FIREWALL-MIB	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 2401	*Security Architecture for the Internet Protocol*
RFC 2402	*IP Authentication Header*
RFC 2428	*FTP Extensions for IPv6 and NATs*
RFC 2460	*Internet Protocol, Version 6 (IPv6) Specification*
RFC 2474	*Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*
RFC 3576	*Change of Authorization*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing Traffic Filters and Firewalls for IPv6 Security

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 40 Feature Information for Implementing Traffic Filters and Firewalls for IPv6 Security

Feature Name	Releases	Feature Information
ACL--Hardware and Software Counters Granularity for IPv4 and IPv6 ACL Statistics	12.2(50)SY	This feature allows both IPv4 and IPv6 ACLs on the Cisco Catalyst 6500 platform to update the ACL entry statistics with a platform entry count.
IOS Zone-Based Firewall	15.1(2)T	Cisco IOS Zone-Based Firewall for IPv6 coexists with Cisco IOS Zone-Based Firewall for IPv4 in order to support IPv6 traffic.
IPv6 ACL Extensions for IPsec Authentication Header	12.4(20)T	The IPv6 ACL extensions for IPsec authentication headers feature allows TCP or UDP parsing when an IPv6 IPsec authentication header is present.
IPv6 Services--Extended Access Control Lists[3]	12.0(23)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(17a)SX112.2(13)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	Standard IPv6 ACL functionality was extended to support traffic filtering based on IPv6 option headers and optional, upper-layer protocol type information for finer granularity of control.
IPv6 Services--IPv6 IOS Firewall	12.3(7)T 12.4 12.4(2)T	This feature provides advanced traffic filtering functionality as an integral part of a network's firewall.
IPv6 Services--IPv6 IOS Firewall FTP Application Support	12.3(11)T 12.4 12.4(2)T	IPv6 supports this feature.

3 IPv6 extended access control lists and IPv6 provider edge router over Multiprotocol Label Switching (MPLS) are implemented with hardware acceleration on the Cisco 12000 series Internet router IP service engine (ISE) line cards in Cisco IOS routers in Cisco IOS Release 12.0(25)S and later releases.

Feature Name	Releases	Feature Information
IPv6 Services--Standard Access Control Lists	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(17a)SX1 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	Access lists determine what traffic is blocked and what traffic is forwarded at router interfaces and allow filtering based on source and destination addresses, inbound and outbound to a specific interface.

Implementing Static Routes for IPv6

This module describes how to configure static routes for IPv6. Routing defines the paths over which packets travel in the network. Manually configured static routes may be used instead of dynamic routing protocols for smaller networks or for sections of a network that have only one path to an outside network. Lack of redundancy limits the usefulness of static routes, and in larger networks manual reconfiguration of routes can become a large administrative overhead.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for IPv6 Routing: Static Routing

You should not configure static configurations over dynamic interfaces, because static configurations will be lost during reboot or when the user disconnects and reconnects the device.

Information About Implementing Static Routes for IPv6

- Floating Static Routes, page 837

Static Routes

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.

Directly Attached Static Routes

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. This example shows such a definition:

```
ipv6 route 2001:DB8::/32 ethernet1/0
```

The example specifies that all destinations with address prefix 2001:DB8::/32 are directly reachable through interface Ethernet1/0.

Directly attached static routes are candidates for insertion in the IPv6 routing table only if they refer to a valid IPv6 interface; that is, an interface that is both up and has 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. This example shows such a definition:

```
ipv6 route 2001:DB8::/32 2001:DB8:3000:1
```

This example specifies that all destinations with address prefix 2001:DB8::/32 are reachable via the host with address 2001:DB8:3000:1.

A recursive static route is valid (that is, it is a candidate for insertion in the IPv6 routing table) only when the specified next hop resolves, either directly or indirectly, to a valid IPv6 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. For example, suppose we have the following routes in the IPv6 routing table:

```
IPv6 Routing Table - 9 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
```

```
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
R   2001:DB8::/32 [130/0]
     via ::, Serial2/0
B   2001:DB8:3000:0/16 [200/45]
     Via 2001:DB8::0104
```

The following examples defines a recursive IPv6 static route:

```
ipv6 route
2001:DB8::/32 2001:0BD8:3000:1
```

This static route will not be inserted into the IPv6 routing table because it is self-recursive. The next hop of the static route, 2001:DB8:3000:1, resolves via the BGP route 2001:DB8:3000:0/16, which is itself a recursive route (that is, it only specifies a next hop). The next hop of the BGP route, 2001:DB8::0104, resolves via 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 IPv6 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 IPv6 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 will be reinserted in the IPv6 routing table.

Note In Cisco IOS Release 12.2(15)T and older releases, IPv6 recursive static routes are checked at one-minute intervals. Therefore, a recursive static route may take up to a minute to be inserted into the routing table once its next hop becomes valid. Likewise, it may take a minute or so for the route to disappear from the table if its next hop becomes invalid.

Fully Specified Static Routes

In a fully specified static route, both the output interface and the next hop are specified. This form of static route is used when the output interface is a multi-access one 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:

```
ipv6 route 2001:DB8:/32 ethernet1/0 2001:DB8:3000:1
```

A fully specified route is valid (that is, a candidate for insertion into the IPv6 routing table) when the specified IPv6 interface is IPv6-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 used in preference to the floating static route. If the dynamic route learned through the routing protocol is lost, the floating static route will be used in its place. The following example defines a floating static route:

```
ipv6 route 2001:DB8:/32 ethernet1/0 2001:DB8:3000:1 210
```

Any of the three types of IPv6 static routes can be used as a floating static route. A floating static route must be configured with an administrative distance that is greater than the administrative distance of the dynamic routing protocol, because routes with smaller administrative distances are preferred.

Note By default, static routes have smaller administrative distances than dynamic routes, so static routes will be used in preference to dynamic routes.

How to Implement Static Routes for IPv6

Configuring a Static IPv6 Route

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 route** *ipv6-prefix / prefix-length ipv6-address | interface-type interface-number ipv6-address*]} [*administrative-distance*] [*administrative-multicast-distance* | **unicast**| **multicast**] [**tag** *tag*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 route** *ipv6-prefix / prefix-length ipv6-address	interface-type interface-number ipv6-address*]} [*administrative-distance*] [*administrative-multicast-distance*	**unicast**| **multicast**] [**tag** *tag* **Example:** `Router(config)# ipv6 route ::/0 serial 2/0`	Configures a static IPv6 route. • A static default IPv6 route is being configured on a serial interface. • See the syntax examples that immediately follow this table for specific uses of the **ipv6 route** command for configuring static routes.

Configuring a Recursive IPv6 Static Route to Use a Default IPv6 Static Route

By default, a recursive IPv6 static route will not resolve using the default route (::/0). Perform this task to restore legacy behavior and allow resolution using the default route.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 route static resolve default**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 route static resolve default** **Example:** `Router(config)# ipv6 route static resolve default`	Allows a recursive IPv6 static route to resolve using the default IPv6 static route.

Configuring a Floating Static IPv6 Route

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 route** *ipv6-prefix / prefix-length* {*ipv6-address* | *interface-type interface-number ipv6-address*]} [*administrative-distance*] [*administrative-multicast-distance* | **unicast** | **multicast**] [**tag** *tag*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 route** *ipv6-prefix* / *prefix-length* {*ipv6-address* | *interface-type interface-number ipv6-address*]} [*administrative-distance*] [*administrative-multicast-distance* | **unicast** | **multicast**] [**tag** *tag* **Example:** `Router(config)# ipv6 route 2001:DB8::/32 serial 2/0 201`	Configures a static IPv6 route. • In this example, a floating static IPv6 route is being configured. An administrative distance of 200 is configured. • Default administrative distances are as follows: ◦ Connected interface--0 ◦ Static route--1 ◦ Enhanced Interior Gateway Routing Protocol (EIGRP) summary route--5 ◦ External Border Gateway Protocol (eBGP)--20 ◦ Internal Enhanced IGRP--90 ◦ IGRP--100 ◦ Open Shortest Path First--110 ◦ Intermediate System-to-Intermediate System (IS-IS)--115 ◦ Routing Information Protocol (RIP)--120 ◦ Exterior Gateway Protocol (EGP)--140 ◦ EIGRP external route--170 ◦ Internal BGP--200 ◦ Unknown--255

Verifying Static IPv6 Route Configuration and Operation

SUMMARY STEPS

1. **enable**
2. Do one of the following:
   - **show ipv6 static** [*ipv6-address* | *ipv6-prefix* / *prefix-length*][**interface** *interface-type interface-number*] [**recursive**] [**detail**]
   -
   -
   - **show ipv6 route** [*ipv6-address* | *ipv6-prefix* / *prefix-length* | *protocol* | *interface-type interface-number*
3. **debug ipv6 routing**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

Command or Action	Purpose

Step 2 Do one of the following:
• **show ipv6 static** [*ipv6-address* | *ipv6-prefix* / *prefix-length*][**interface** *interface-type interface-number*] [**recursive**] [**detail**]
•
•
• **show ipv6 route** [*ipv6-address* | *ipv6-prefix* / *prefix-length* | *protocol* | *interface-type interface-number*
Example:
`Router# show ipv6 static`
Example:
Example:
Example:
`Router# show ipv6 route static`
	Displays the current contents of the IPv6 routing table.
	• These examples show two different ways of displaying IPv6 static routes.

Step 3 **debug ipv6 routing** **Example:** `Router# debug ipv6 routing`	Displays debugging messages for IPv6 routing table updates and route cache updates.

• Examples, page 842

Examples

Sample Output from the ipv6 route Command

The following example shows how to configure a directly attached static route through a point-to-point interface.

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

The following example shows how to configure a directly attached static route on a broadcast interface.

```
Router(config)# ipv6 route 2001:DB8::1/32 ethernet1/0
```

The following example shows how to configure a fully specified static route on a broadcast interface.

```
Router(config)# ipv6 route 2001:DB8::1/32 ethernet1/0 fe80::1
```

In the following example, a static route is being configured to a specified next-hop address, from which the output interface is automatically derived.

```
Router(config)# ipv6 route 2001:DB8::/32 2001:DB8:2002:1>>
```

Sample Output from the show ipv6 static Command when No Options Are Specified in the Command Syntax

When no options are specified in the command, those routes installed in the IPv6 routing table are marked with an asterisk, as shown in the following example:

```
Router# show ipv6 static
IPv6 Static routes
Code: * - installed in RIB
* 2001:DB8:3000:0/16, interface Ethernet1/0, distance 1
* 2001:DB8:4000:0/16, via nexthop 2001:DB8:1:1, distance 1
  2001:DB8:5000:0/16, interface Ethernet3/0, distance 1
* 2001:DB8:5555:0/16, via nexthop 2001:DB8:4000:1, distance 1
  2001:DB8:5555:0/16, via nexthop 2001:DB8:9999:1, distance 1
* 2001:DB8:5555:0/16, interface Ethernet2/0, distance 1
* 2001:DB8:6000:0/16, via nexthop 2001:DB8:2007:1, interface Ethernet1/0, distance 1
```

Sample Output from the show ipv6 static Command with the IPv6 Address and Prefix Command

When the *ipv6-address* or *ipv6-prefix* / *prefix-length* argument is specified, only information about static routes for that address or network is displayed. The following is sample output from the **show ipv6 static**command when entered with the IPv6 prefix 2001:DB8:200::/35:

```
Router# show ipv6 static 2001:DB8:5555:0/16
IPv6 Static routes
Code: * - installed in RIB
* 2001:DB8:5555:0/16, via nexthop 2001:DB8:4000:1, distance 1
  2001:DB8:5555:0/16, via nexthop 2001:9999:1, distance 2
* 2001:DB8:5555:0/16, interface Ethernet2/0, distance 1
```

Sample Output from the show ipv6 static interface Command

When an interface is supplied, only those static routes with the specified interface as outgoing interface are displayed. The **interface** keyword may be used with or without the IPv6 address and prefix specified in the **show ipv6 static** command.

```
Router# show ipv6 static interface ethernet3/0
IPv6 Static routes
Code: * - installed in RIB
```

Sample Output from the show ipv6 static recursive Command

When the **recursive** keyword is specified in the **show ipv6 static** command, only recursive static routes are displayed. The **recursive** keyword is mutually exclusive with the **interface** keyword, but it may be used with or without the IPv6 prefix included in the command syntax.

```
Router# show ipv6 static recursive
IPv6 Static routes
Code: * - installed in RIB
* 2001:DB8:4000:0/16, via nexthop 2001:DB8:1:1, distance 1
* 2001:DB8:5555:0/16, via nexthop 2001:DB8:4000:1, distance 2
  2001:DB8:5555:0/16, via nexthop 2001:DB8:9999:1, distance 3
```

Sample Output from the show ipv6 static detail Command

When the **detail** keyword is specified, the following additional information is also displayed:

- For valid recursive routes, the output path set, and maximum resolution depth
- For invalid recursive routes, the reason why the route is not valid.
- For invalid direct or fully specified routes, the reason why the route is not valid.

```
Router# show ipv6 static detail
IPv6 Static routes
Code: * - installed in RIB
* 2001:DB8:3000:0/16, interface Ethernet1/0, distance 1
* 2001:DB8:4000:0/16, via nexthop 2001:DB8:2001:1, distance 1
     Resolves to 1 paths (max depth 1)
     via Ethernet1/0
  2001:DB8:5000:0/16, interface Ethernet3/0, distance 1
     Interface is down
* 2001:DB8:5555:0/16, via nexthop 2001:DB8:4000:1, distance 1
     Resolves to 1 paths (max depth 2)
     via Ethernet1/0
  2001:DB8:5555:0/16, via nexthop 2001:DB8:9999:1, distance 1
     Route does not fully resolve
* 2001:DB8:5555:0/16, interface Ethernet2/0, distance 1
* 2001:DB8:6000:0/16, via nexthop 2001:DB8:2007:1, interface Ethernet1/0, distance 1
```

Sample Output from the show ipv6 route Command

In the following example, the **show ipv6 route** command is used to verify the configuration of a static route through a point-to-point interface:

```
Router# show ipv6 route
IPv6 Routing Table - 9 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
S   2001:DB8::/32 [1/0]
     via ::, Serial2/0
```

In the following example, the **show ipv6 route** command is used to verify the configuration of a static route on a multiaccess interface. An IPv6 link-local address--FE80::1--is the next-hop router.

```
Router# show ipv6 route
IPv6 Routing Table - 11 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
S   2001:DB8::/32 [1/0]
     via FE80::1, Ethernet0/0
```

To display all static routes in the IPv6 routing table, use the **show ipv6 route static** command is used with static as the value of the protocol argument:

```
Router# show ipv6 route static
IPv6 Routing Table - 330 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
S   2001:DB8::/32 [1/0]
     via ::, Tunnel0
S   3FFE:C00:8011::/48 [1/0]
     via ::, Null0
S   ::/0 [254/0]
     via 2001:DB8:2002:806B, Null
```

Sample Output from the debug ipv6 routing Command

In the following example, the **debug ipv6 routing** command is used to verify the installation of a floating static route into the IPv6 routing table when an IPv6 RIP route is deleted. The floating static IPv6 route was previously configured with an administrative distance value of 130. The backup route was added as a floating static route because RIP routes have a default administrative distance of 120, and the RIP route should be the preferred route. When the RIP route is deleted, the floating static route is installed in the IPv6 routing table.

```
Router# debug ipv6 routing
*Oct 10 18:28:00.847: IPv6RT0: rip two, Delete 2001:DB8::/32 from table
*Oct 10 18:28:00.847: IPv6RT0: static, Backup call for 2001:DB8::/32
*Oct 10 18:28:00.847: IPv6RT0: static, Add 2001:DB8::/32 to table
*Oct 10 18:28:00.847: IPv6RT0: static, Adding next-hop :: over Serial2/0 for
2001:DB8::/32, [130/0]
```

Configuration Examples for Implementing Static Routes for IPv6

Static routes may be used for a variety of purposes. Common usages include the following:

- Manual summarization
- Traffic discard
- Fixed default route
- Backup route

In many cases, alternative mechanisms exist within Cisco IOS software to achieve the same objective. Whether to use static routes or one of the alternative mechanisms depends on local circumstances.

- Example: Configuring Manual Summarization, page 845
- Example: Configuring Traffic Discard, page 846
- Example: Configuring a Fixed Default Route, page 846
- Example: Configuring a Floating Static Route, page 846

Example: Configuring Manual Summarization

The following example shows a static route being used to summarize local interface prefixes advertised into RIP. The static route also serves as a discard route, discarding any packets received by the router to a 2001:DB8:1::/48 destination not covered by a more specific interface prefix.

```
Router> enable
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# interface ethernet0/0
Router(config-if)# ipv6 address 2001:DB8:2:1234/64
Router(config-if)# exit
Router(config)#
Router(config)# interface ethernet1/0
Router(config-if)# ipv6 address 2001:DB8:3:1234/64
Router(config-if)# exit
Router(config)# interface ethernet2/0
Router(config-if)# ipv6 address 2001:DB8:4:1234/64
Router(config-if)# exit
Router(config)# interface ethernet3/0
Router(config-if)# ipv6 address 2001:DB8::1234/64
Router(config-if)# ipv6 rip one enable
Router(config-if)# exit
```

```
Router(config)# ipv6 router rip one
Router(config-rtr)# redistribute static
Router(config-rtr)# exit
Router(config)# ipv6 route 2001:DB8:1:1/48 null0
Router(config)# end
00:01:30: %SYS-5-CONFIG_I: Configured from console by console
Router# show ipv6 route static

IPv6 Routing Table - 3 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
S   2001:DB8:1::/48 [1/0]
     via ::, Null0
```

Example: Configuring Traffic Discard

Configuring a static route to point at interface null0 may be used for discarding traffic to a particular prefix. For example, if it is required to discard all traffic to prefix 2001:DB8:42:1/64, the following static route would be defined:

```
Router> enable
Router# configure
 terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# ipv6 route 2001:DB8:42:1::/64 null0
Router(config)# end
```

Example: Configuring a Fixed Default Route

A default static route is often used in simple router topologies. In the following example, a router is connected to its local site via Ethernet0/0 and to the main corporate network via Serial2/0 and Serial3/0. All nonlocal traffic will be routed over the two serial interfaces.

```
Router(config)# interface ethernet0/0
Router(config-if)# ipv6 address 2001:DB8:17:1234/64
Router(config-if)# exit
Router(config)# interface Serial2/0
Router(config-if)# ipv6 address 2001:DB8:1:1234/64
Router(config-if)# exit
Router(config)# interface Serial3/0
Router(config-if)# ipv6 address 2001:DB8:2:124/64
Router(config-if)# exit
Router(config)# ipv6 route ::/0 Serial2/0
Router(config)# ipv6 route ::/0 Serial3/0
Router(config)# end
Router#
00:06:30: %SYS-5-CONFIG_I: Configured from console by console
Router# show ipv6 route static
IPv6 Routing Table - 7 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
S   ::/0 [1/0]
     via ::, Serial2/0
     via ::, Serial3/0
```

Example: Configuring a Floating Static Route

A floating static route often is used to provide a backup path in the event of connectivity failure. In the following example, the router has connectivity to the network core via Serial2/0 and learns the route

2001:DB8:1:1/32 via IS-IS. If the Serial2/0 interface fails, or if route 2001:DB8:1:1/32 is no longer learned via IS-IS (indicating loss of connectivity elsewhere in the network), traffic is routed via the backup ISDN interface.

```
Router> enable
Router# configure
 terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# interface ethernet0/0
Router(config-if)# ipv6 address 2001:DB8:17:1234/64
Router(config-if)# exit
Router(config)# interface Serial2/0
Router(config-if)# ipv6 address 2001:DB8:1:1234/64
Router(config-if)# ipv6 router isis
Router(config-if)# exit
Router(config)# router isis
Router(config-router)# net 42.0000.0000.0000.0001.00
Router(config-router)# exit
Router(config)# interface BRI1/0
Router(config-if)# encapsulation ppp
Router(config-if)# ipv6 enable
Router(config-if)# isdn switch-type basic-net3
Router(config-if)# ppp authentication chap optional
Router(config-if)# ppp multilink
Router(config-if)# exit
Router(config)# dialer-list 1 protocol ipv6 permit
Router(config)# ipv6 route 2001:DB8:1::/32 BRI1/0 200
Router(config)# end
Router#
00:03:07: %SYS-5-CONFIG_I: Configured from console by console
2001:DB8:5000:)/16, interface Ethernet3/0, distance 1
```

Additional References

Related Documents

Related Topic	Document Title
IP static route configuration	" Protocol-Independent Routing," *Cisco IOS IP Routing Protocols Configuration Guide*
IP static route commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IP Routing Protocols Command Reference*
IPv6 supported feature list	" Start Here: Cisco IOS Software Release Specifics for IPv6 Features ," *Cisco IOS IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIBs	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing Static Routes for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 41 Feature Information for Implementing Static Routes for IPv6

Feature Name	Releases	Feature Information
IPv6 Routing--Static Routing	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(17a)SX1 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	Static routes are manually configured and define an explicit path between two networking devices.

Implementing Tunneling for IPv6

This module describes how to configure overlay tunneling techniques used by the Cisco IOS software to support the transition from IPv4-only networks to integrated IPv4- and IPv6-based networks. Tunneling encapsulates IPv6 packets in IPv4 packets and uses the IPv4 network as a link-layer mechanism.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Implementing Tunneling for IPv6

- In Cisco IOS Release 12.0(21)ST and Cisco IOS Release 12.0(22)S and earlier releases, the Cisco 12000 series Gigabit Switch Router (GSR) gives a very low priority to the processing of IPv6 tunneled packets. Therefore, we strongly recommend that you limit the use of IPv6 tunnels on the GSR using these releases to topologies that sustain a low level of network traffic and require a minimal amount of process-switching resources.
- IPv6 manually configured tunnel traffic in Cisco IOS Release 12.0(23)S is processed in software on the CPU of the line card, instead of in the Route Processor (RP) in the GSR, resulting in enhanced performance.

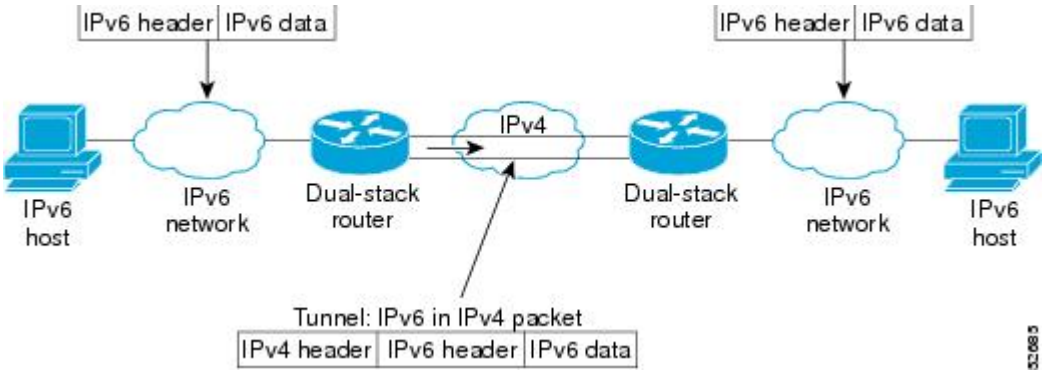
Information About Implementing Tunneling for IPv6

Overlay Tunnels for IPv6

Overlay tunneling encapsulates IPv6 packets in IPv4 packets for delivery across an IPv4 infrastructure (a core network or the Internet (see the figure below). By using overlay tunnels, you can communicate with isolated IPv6 networks without upgrading the IPv4 infrastructure between them. Overlay tunnels can be configured between border routers or between a border router and a host; however, both tunnel endpoints must support both the IPv4 and IPv6 protocol stacks. IPv6 supports the following types of overlay tunneling mechanisms:

- Manual
- Generic routing encapsulation (GRE)
- IPv4-compatible
- 6to4
- Intrasite Automatic Tunnel Addressing Protocol (ISATAP)

Figure 54 Overlay Tunnels



Note Overlay tunnels reduce the maximum transmission unit (MTU) of an interface by 20 octets (assuming the basic IPv4 packet header does not contain optional fields). A network using overlay tunnels is difficult to troubleshoot. Therefore, overlay tunnels connecting isolated IPv6 networks should not be considered as a final IPv6 network architecture. The use of overlay tunnels should be considered as a transition technique toward a network that supports both the IPv4 and IPv6 protocol stacks or just the IPv6 protocol stack.

Use the table below to help you determine which type of tunnel you want to configure to carry IPv6 packets over an IPv4 network.

Table 42 Suggested Usage of Tunnel Types to Carry IPv6 Packets over an IPv4 Network

Tunneling Type	Suggested Usage	Usage Notes
Manual	Simple point-to-point tunnels that can be used within a site or between sites	Can carry IPv6 packets only.
GRE- and IPv4- compatible	Simple point-to-point tunnels that can be used within a site or between sites	Can carry IPv6, Connectionless Network Service (CLNS), and many other types of packets.
IPv4- compatible	Point-to-multipoint tunnels	Uses the ::/96 prefix. We do not now recommend using this tunnel type.
6to4	Point-to-multipoint tunnels that can be used to connect isolated IPv6 sites	Sites use addresses from the 2002::/16 prefix.
6RD	IPv6 service is provided to customers over an IPv4 network by using encapsulation of IPv6 in IPv4.	Prefixes can be from the SP's own address block.
ISATAP	Point-to-multipoint tunnels that can be used to connect systems within a site	Sites can use any IPv6 unicast addresses.

Individual tunnel types are discussed in detail in this document. We recommend that you review and understand the information about the specific tunnel type that you want to implement. When you are familiar with the type of tunnel you need, see the table below for a summary of the tunnel configuration parameters that you may find useful.

Table 43 Tunnel Configuration Parameters by Tunneling Type

Tunneling Type	Tunnel Configuration Parameter			
Tunnel Mode	Tunnel Source	Tunnel Destination	Interface Prefix or Address	
Manual	ipv6ip	An IPv4 address, or a reference to an interface on which IPv4 is configured.	An IPv4 address.	An IPv6 address.
GRE/IPv4	gre ip		An IPv4 address.	An IPv6 address.

Tunneling Type	Tunnel Configuration Parameter			
IPv4- compatible	ipv6ip auto-tunnel		Not required. These are all point-to-multipoint tunneling types. The IPv4 destination address is calculated, on a per-packet basis, from the IPv6 destination.	Not required. The interface address is generated as ::*tunnel-source*/96.
6to4	ipv6ip 6to4			An IPv6 address. The prefix must embed the tunnel source IPv4 address
6RD	ipv6ip 6rd			An IPv6 address.
ISATAP	ipv6ip isatap			An IPv6 prefix in modified eui-64 format. The IPv6 address is generated from the prefix and the tunnel source IPv4 address.

IPv6 Manually Configured Tunnels

A manually configured tunnel is equivalent to a permanent link between two IPv6 domains over an IPv4 backbone. The primary use is for stable connections that require regular secure communication between two edge routers or between an end system and an edge router, or for connection to remote IPv6 networks.

An IPv6 address is manually configured on a tunnel interface, and manually configured IPv4 addresses are assigned to the tunnel source and the tunnel destination. The host or router at each end of a configured tunnel must support both the IPv4 and IPv6 protocol stacks. Manually configured tunnels can be configured between border routers or between a border router and a host. Cisco Express Forwarding switching can be used for IPv6 manually configured tunnels, or Cisco Express Forwarding switching can be disabled if process switching is needed.

GRE IPv4 Tunnel Support for IPv6 Traffic

IPv6 traffic can be carried over IPv4 GRE tunnels using the standard GRE tunneling technique that is designed to provide the services necessary to implement any standard point-to-point encapsulation scheme. As in IPv6 manually configured tunnels, GRE tunnels are links between two points, with a separate tunnel for each link. The tunnels are not tied to a specific passenger or transport protocol, but in this case, carry IPv6 as the passenger protocol with the GRE as the carrier protocol and IPv4 or IPv6 as the transport protocol.

The primary use of GRE tunnels is for stable connections that require regular secure communication between two edge routers or between an edge router and an end system. The edge routers and the end systems must be dual-stack implementations.

GRE Support over IPv6 Transport

GRE has a protocol field that identifies the passenger protocol. GRE tunnels allow Intermediate System-to-Intermediate System (IS-IS) or IPv6 to be specified as a passenger protocol, which allows both IS-IS and IPv6 traffic to run over the same tunnel. If GRE did not have a protocol field, it would be impossible to distinguish whether the tunnel was carrying IS-IS or IPv6 packets. The GRE protocol field is why it is desirable that you tunnel IS-IS and IPv6 inside GRE.

mGRE Tunnels Support over IPv6

To enable service providers deploy IPv6 in their core infrastructure, multipoint generic routing encapsulation (mGRE) tunnels over IPv6 are supported. Dynamic Multipoint Virtual Private Network (DMVPN) customers may run either IPv4 or IPv6 in their local networks, so the overlay endpoints can be either IPv4 or IPv6. For an IPv6 transport endpoint, the overlay endpoint can either be an IPv4 or IPv6 private network address.

GRE has a protocol field that identifies the passenger protocol. GRE tunnels allow Intermediate System-to-Intermediate System (IS-IS) or IPv6 to be specified as a passenger protocol, which allows both IS-IS and IPv6 traffic to run over the same tunnel. If GRE did not have a protocol field, it would be impossible to distinguish whether the tunnel was carrying IS-IS or IPv6 packets. The GRE protocol field is why it is desirable that you tunnel IS-IS and IPv6 inside GRE.

GRE CLNS Tunnel Support for IPv4 and IPv6 Packets

GRE tunneling of IPv4 and IPv6 packets through CLNS networks enables Cisco CLNS Tunnels (CTunnels) to interoperate with networking equipment from other vendors. This feature provides compliance with RFC 3147.

The optional GRE services defined in header fields, such as checksums, keys, and sequencing, are not supported. Any packet received requesting such services will be dropped.

Automatic 6to4 Tunnels

An automatic 6to4 tunnel allows isolated IPv6 domains to be connected over an IPv4 network to remote IPv6 networks. The key difference between automatic 6to4 tunnels and manually configured tunnels is that the tunnel is not point-to-point; it is point-to-multipoint. In automatic 6to4 tunnels, routers are not configured in pairs because they treat the IPv4 infrastructure as a virtual nonbroadcast multiaccess (NBMA) link. The IPv4 address embedded in the IPv6 address is used to find the other end of the automatic tunnel.

An automatic 6to4 tunnel may be configured on a border router in an isolated IPv6 network, which creates a tunnel on a per-packet basis to a border router in another IPv6 network over an IPv4 infrastructure. The tunnel destination is determined by the IPv4 address of the border router extracted from the IPv6 address that starts with the prefix 2002::/16, where the format is 2002:*border-router-IPv4-address* ::/48. Following the embedded IPv4 address are 16 bits that can be used to number networks within the site. The border router at each end of a 6to4 tunnel must support both the IPv4 and IPv6 protocol stacks. 6to4 tunnels are configured between border routers or between a border router and a host.

The simplest deployment scenario for 6to4 tunnels is to interconnect multiple IPv6 sites, each of which has at least one connection to a shared IPv4 network. This IPv4 network could be the global Internet or a corporate backbone. The key requirement is that each site have a globally unique IPv4 address; the Cisco software uses this address to construct a globally unique 6to4/48 IPv6 prefix. As with other tunnel

mechanisms, appropriate entries in a Domain Name System (DNS) that map between hostnames and IP addresses for both IPv4 and IPv6 allow the applications to choose the required address.

Automatic IPv4-Compatible IPv6 Tunnels

Automatic IPv4-compatible tunnels use IPv4-compatible IPv6 addresses. IPv4-compatible IPv6 addresses are IPv6 unicast addresses that have zeros in the high-order 96 bits of the address, and an IPv4 address in the low-order 32 bits. They can be written as 0:0:0:0:0:0:A.B.C.D or ::A.B.C.D, where "A.B.C.D" represents the embedded IPv4 address.

The tunnel destination is automatically determined by the IPv4 address in the low-order 32 bits of IPv4-compatible IPv6 addresses. The host or router at each end of an IPv4-compatible tunnel must support both the IPv4 and IPv6 protocol stacks. IPv4-compatible tunnels can be configured between border-routers or between a border-router and a host. Using IPv4-compatible tunnels is an easy method to create tunnels for IPv6 over IPv4, but the technique does not scale for large networks.

IPv6 Rapid Deployment Tunnels

The IPv6 Rapid Deployment (6RD) feature is an extension of the 6to4 feature. The 6RD feature allows a service provider to provide a unicast IPv6 service to customers over its IPv4 network by using encapsulation of IPv6 in IPv4.

The main differences between 6RD and 6to4 tunneling are as follows:

- 6RD does not require addresses to have a 2002::/16 prefix; therefore, the prefix can be from theservice provider’s own address block. This function allows the 6RD operational domain to be within the SP network. From the perspective of customer sites and the general IPv6 Internet connected to a 6RD-enabled service provider network, the IPv6 service provided is equivalent to the native IPv6.
- All 32 bits of the IPv4 destination need not be carried in the IPv6 payload header. The IPv4 destination is obtained from a combination of bits in the payload header and information on the router. Furthermore, the IPv4 address is not at a fixed location in the IPv6 header as it is in 6to4.

ISATAP Tunnels

ISATAP is an automatic overlay tunneling mechanism that uses the underlying IPv4 network as an NBMA link layer for IPv6. ISATAP is designed for transporting IPv6 packets within a site where a native IPv6 infrastructure is not yet available; for example, when sparse IPv6 hosts are deployed for testing. ISATAP tunnels allow individual IPv4 or IPv6 dual-stack hosts within a site to communicate with other such hosts on the same virtual link, basically creating an IPv6 network using the IPv4 infrastructure.

The ISATAP router provides standard router advertisement network configuration support for the ISATAP site. This feature allows clients to automatically configure themselves as they would do if they were connected to an Ethernet. It can also be configured to provide connectivity out of the site. ISATAP uses a well-defined IPv6 address format composed of any unicast IPv6 prefix (/64), which can be link local, or global (including 6to4 prefixes), enabling IPv6 routing locally or on the Internet. The IPv4 address is encoded in the last 32 bits of the IPv6 address, enabling automatic IPv6-in-IPv4 tunneling.

Although the ISATAP tunneling mechanism is similar to other automatic tunneling mechanisms, such as IPv6 6to4 tunneling, ISATAP is designed for transporting IPv6 packets within a site, but not between sites.

Although the ISATAP tunneling mechanism is similar to other automatic tunneling mechanisms, such as IPv6 6to4 tunneling, ISATAP is designed for transporting IPv6 packets within a site, not between sites.

ISATAP uses unicast addresses that include a 64-bit IPv6 prefix and a 64-bit interface identifier. The interface identifier is created in modified EUI-64 format in which the first 32 bits contain the value

000:5EFE to indicate that the address is an IPv6 ISATAP address. The table below describes an ISATAP address format.

Table 44 IPv6 ISATAP Address Format

64 Bits	32 Bits	32 Bits
Link local or global IPv6 unicast prefix	0000:5EFE	IPv4 address of the ISATAP link

As shown in the table above, an ISATAP address consists of an IPv6 prefix and the ISATAP interface identifier. This interface identifier includes the IPv4 address of the underlying IPv4 link. The following example shows what an actual ISATAP address would look like if the prefix is 2001:DB8:1234:5678::/64 and the embedded IPv4 address is 10.173.129.8. In the ISATAP address, the IPv4 address is expressed in hexadecimal as 0AAD:8108 (for example, 2001:DB8:1234:5678:0000:5EFE:0AAD:8108).

IPv6 IPsec Site-to-Site Protection Using Virtual Tunnel Interface

The IPv6 IPsec feature provides IPv6 crypto site-to-site protection of all types of IPv6 unicast and multicast traffic using native IPsec IPv6 encapsulation. The IPsec virtual tunnel interface (VTI) feature provides this function, using IKE as the management protocol.

An IPsec VTI supports native IPsec tunneling and includes most of the properties of a physical interface. The IPsec VTI alleviates the need to apply crypto maps to multiple interfaces and provides a routable interface.

The IPsec VTI allows IPv6 routers to work as security gateways, establish IPsec tunnels between other security gateway routers, and provide crypto IPsec protection for traffic from internal network when being transmitting across the public IPv6 Internet.

For further information on VTIs, see Implementing IPsec in IPv6 Security.

How to Implement Tunneling for IPv6

Configuring Manual IPv6 Tunnels

Perform this task to configure manual IPv6 tunnels.

With manually configured IPv6 tunnels, an IPv6 address is configured on a tunnel interface, and manually configured IPv4 addresses are assigned to the tunnel source and the tunnel destination. The host or router at each end of a configured tunnel must support both the IPv4 and IPv6 protocol stacks.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *tunnel-number*
4. **ipv6 address** *ipv6-prefix / prefix-length* [**eui-64**]
5. **tunnel source** {*ip-address*| *interface-t ype interface-number*}
6. **tunnel destination** *ip-address*
7. **tunnel mode ipv6ip**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface tunnel** *tunnel-number* **Example:** `Router(config)# interface tunnel 0`	Specifies a tunnel interface and number, and enters interface configuration mode.
Step 4	**ipv6 address** *ipv6-prefix / prefix-length* [**eui-64**] **Example:** `Router(config-if)# ipv6 address 3ffe:b00:c18:1::3/127`	Specifies the IPv6 network assigned to the interface and enables IPv6 processing on the interface.
Step 5	**tunnel source** {*ip-address*| *interface-t ype interface-number*} **Example:** `Router(config-if)# tunnel source ethernet 0`	Specifies the source IPv4 address or the source interface type and number for the tunnel interface. • If an interface is specified, the interface must be configured with an IPv4 address.

	Command or Action	Purpose
Step 6	**tunnel destination** *ip-address* **Example:** `Router(config-if)# tunnel destination 192.168.30.1`	Specifies the destination IPv4 address or hostname for the tunnel interface.
Step 7	**tunnel mode ipv6ip** **Example:** `Router(config-if)# tunnel mode ipv6ip`	Specifies a manual IPv6 tunnel. **Note** The **tunnel mode ipv6ip** command specifies IPv6 as the passenger protocol and IPv4 as both the encapsulation and transport protocol for the manual IPv6 tunnel.

Configuring GRE IPv6 Tunnels

Perform this task to configure a GRE tunnel on an IPv6 network. GRE tunnels can be configured to run over an IPv6 network layer and to transport IPv6 packets in IPv6 tunnels and IPv4 packets in IPv6 tunnels.

When GRE IPv6 tunnels are configured, IPv6 addresses are assigned to the tunnel source and the tunnel destination. The tunnel interface can have either IPv4 or IPv6 addresses assigned (this is not shown in the task). The host or router at each end of a configured tunnel must support both the IPv4 and IPv6 protocol stacks.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *tunnel-number*
4. **ipv6 address** *ipv6-prefix / prefix-length* [**eui-64**]
5. **tunnel source** {*ip-address* | *ipv6-address* | *interface-type interface-number*}
6. **tunnel destination** {*host-name* | *ip-address* | *ipv6-address*}
7. **tunnel mode** {**aurp** | **cayman** | **dvmrp** | **eon** | **gre**| **gre multipoint** | **gre ipv6** | **ipip** [**decapsulate-any**] | **iptalk** | **ipv6** | **mpls** | **nos**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface tunnel** *tunnel-number* **Example:** `Router(config)# interface tunnel 0`	Specifies a tunnel interface and number, and enters interface configuration mode.
Step 4	**ipv6 address** *ipv6-prefix* / *prefix-length* [**eui-64**] **Example:** `Router(config-if)# ipv6 address 3ffe:b00:c18:1::3/127`	Specifies the IPv6 network assigned to the interface and enables IPv6 processing on the interface.
Step 5	**tunnel source** {*ip-address* | *ipv6-address* | *interface-type interface-number*} **Example:** `Router(config-if)# tunnel source ethernet 0`	Specifies the source IPv4 address or the source interface type and number for the tunnel interface. • If an interface is specified, the interface must be configured with an IPv4 address.
Step 6	**tunnel destination** {*host-name* | *ip-address* | *ipv6-address*} **Example:** `Router(config-if)# tunnel destination 2001:DB8:1111:2222::1/64`	Specifies the destination IPv6 address or hostname for the tunnel interface.
Step 7	**tunnel mode** {**aurp** | **cayman** | **dvmrp** | **eon** | **gre**| **gre multipoint** | **gre ipv6** | **ipip** [**decapsulate-any**] | **iptalk** | **ipv6** | **mpls** | **nos** **Example:** `Router(config-if)# tunnel mode gre ipv6`	Specifies a GRE IPv6 tunnel. **Note** The **tunnel mode gre ipv6**command specifies GRE as the encapsulation protocol for the tunnel.

Configuring Automatic 6to4 Tunnels

Perform this task to configure automatic 6to4 tunnels.

With 6to4 tunnels, the tunnel destination is determined by the border router IPv4 address, which is concatenated to the prefix 2002::/16 in the format 2002:*border-router-IPv4-address*::/48. The border router at each end of a 6to4 tunnel must support both the IPv4 and IPv6 protocol stacks.

Note The configuration of only one IPv4-compatible tunnel and one 6to4 IPv6 tunnel is supported on a router. If you choose to configure both of those tunnel types on the same router, we strongly recommend that they do not share the same tunnel source.

The reason that a 6to4 tunnel and an IPv4-compatible tunnel cannot share an interface is that both of them are NBMA "point-to-multipoint" access links and only the tunnel source can be used to reorder the packets from a multiplexed packet stream into a single packet stream for an incoming interface. So when a packet with an IPv4 protocol type of 41 arrives on an interface, that packet is mapped to an IPv6 tunnel interface based on the IPv4 address. However, if both the 6to4 tunnel and the IPv4-compatible tunnel share the same source interface, the router is not able to determine the IPv6 tunnel interface to which it should assign the incoming packet.

IPv6 manually configured tunnels can share the same source interface because a manual tunnel is a "point-to-point" link, and both the IPv4 source and IPv4 destination of the tunnel are defined.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *tunnel-number*
4. **ipv6 address** *ipv6-prefix* / *prefix-length* [**eui-64**]
5. **tunnel source** {*ip-address*| *interface-t ype interface-number*}
6. **tunnel mode ipv6ip 6to4**
7. **exit**
8. **ipv6 route** *ipv6-prefix* / *prefix-length* **tunnel** *tunnel-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface tunnel** *tunnel-number* **Example:** `Router(config)# interface tunnel 0`	Specifies a tunnel interface and number, and enters interface configuration mode.
Step 4	**ipv6 address** *ipv6-prefix* / *prefix-length* [**eui-64**] **Example:** `Router(config-if)# ipv6 address 2002:c0a8:6301:1::1/64`	Specifies the IPv6 address assigned to the interface and enables IPv6 processing on the interface. • The 32 bits following the initial 2002::/16 prefix correspond to an IPv4 address assigned to the tunnel source.
Step 5	**tunnel source** {*ip-address*| *interface-t ype interface-number*} **Example:** `Router(config-if)# tunnel source ethernet 0`	Specifies the source interface type and number for the tunnel interface. **Note** The interface type and number specified in the **tunnel source** command must be configured with an IPv4 address.
Step 6	**tunnel mode ipv6ip 6to4** **Example:** `Router(config-if)# tunnel mode ipv6ip 6to4`	Specifies an IPv6 overlay tunnel using a 6to4 address.
Step 7	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode, and returns the router to global configuration mode.
Step 8	**ipv6 route** *ipv6-prefix* / *prefix-length* **tunnel** *tunnel-number* **Example:** **Example:** `Router(config)# ipv6 route 2002::/16 tunnel 0`	Configures a static route for the IPv6 6to4 prefix 2002::/16 to the specified tunnel interface. **Note** When configuring a 6to4 overlay tunnel, you must configure a static route for the IPv6 6to4 prefix 2002::/16 to the 6to4 tunnel interface. • The tunnel number specified in the **ipv6 route** command must be the same tunnel number specified in the **interface tunnel**command.

Configuring IPv4-Compatible IPv6 Tunnels

Perform this task to configure IPv4-compatible IPv6 tunnels.

With an IPv4-compatible tunnel, the tunnel destination is automatically determined by the IPv4 address in the low-order 32 bits of IPv4-compatible IPv6 addresses. The host or router at each end of an IPv4-compatible tunnel must support both the IPv4 and IPv6 protocol stacks.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *tunnel-number*
4. **tunnel source** {*ip-address*| *interface-t ype interface-number*}
5. **tunnel mode ipv6ip auto-tunnel**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface tunnel** *tunnel-number* **Example:** `Router(config)# interface tunnel 0`	Specifies a tunnel interface and number, and enters interface configuration mode.
Step 4	**tunnel source** {*ip-address*| *interface-t ype interface-number*} **Example:** `Router(config-if)# tunnel source ethernet 0`	Specifies the source interface type and number for the tunnel interface. **Note** The interface type and number specified in the **tunnel source** command is configured with an IPv4 address only.
Step 5	**tunnel mode ipv6ip auto-tunnel** **Example:** `Router(config-if)# tunnel mode ipv6ip auto-tunnel`	Specifies an IPv4-compatible tunnel using an IPv4-compatible IPv6 address.

Configuring 6RD Tunnels

Perform this task to configure 6RD tunnels.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *tunnel-number*
4. **tunnel source** {*ip-address*| *interface-t ype interface-number*}
5. **tunnel mode ipv6ip** [**6rd** | **6to4** | **auto-tunnel** | **isatap**]
6. **tunnel 6rd prefix** *ipv6-prefix* / *prefix-length*
7. **tunnel 6rd ipv4** {**prefix-length** *length*} {**suffix-length** *length*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface tunnel** *tunnel-number* **Example:** `Router(config)# interface tunnel 1`	Specifies a tunnel interface and number, and enters interface configuration mode.
Step 4	**tunnel source** {*ip-address*| *interface-t ype interface-number*} **Example:** `Router(config-if)# tunnel source Ethernet2/0`	Specifies the source interface type and number for the tunnel interface.
Step 5	**tunnel mode ipv6ip** [**6rd**	**6to4**	**auto-tunnel**	**isatap**] **Example:** `Router(config-if)# tunnel mode ipv6ip 6rd`	Configures a static IPv6 tunnel interface.

	Command or Action	Purpose
Step 6	**tunnel 6rd prefix** *ipv6-prefix* / *prefix-length* **Example:** `Router(config-if)# tunnel 6rd prefix 2001:B000::/32`	Specifies the common IPv6 prefix on IPv6 rapid 6RD tunnels.
Step 7	**tunnel 6rd ipv4** {**prefix-length** *length*} {**suffix-length** *length* **Example:** `Router(config-if)# tunnel 6rd ipv4 prefix-length 16 suffix 8`	Specifies the prefix length and suffix length of the IPv4 transport address common to all the 6RD routers in a domain.

Configuring ISATAP Tunnels

The **tunnel source** command used in the configuration of an ISATAP tunnel must point to an interface with an IPv4 address configured. The ISATAP IPv6 address and prefix (or prefixes) advertised are configured as for a native IPv6 interface. The IPv6 tunnel interface must be configured with a modified EUI-64 address because the last 32 bits in the interface identifier are constructed using the IPv4 tunnel source address.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *tunnel-number*
4. **ipv6 address** *ipv6-prefix* / *prefix-length* [**eui-64**]
5. **no ipv6 nd ra suppress**
6. **tunnel source** {*ip-address*| *interface-type interface-number*}
7. **tunnel mode ipv6ip isatap**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

Command or Action	Purpose
Step 2 **configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3 **interface tunnel** *tunnel-number* **Example:** `Router(config)# interface tunnel 1`	Specifies a tunnel interface and number, and enters interface configuration mode.
Step 4 **ipv6 address** *ipv6-prefix* / *prefix-length* [**eui-64**] **Example:** `Router(config-if)# ipv6 address 2001:DB8:6301::/64 eui-64`	Specifies the IPv6 address assigned to the interface and enables IPv6 processing on the interface. **Note** Refer to the *Configuring Basic Connectivity for IPv6* module for more information on configuring IPv6 addresses.
Step 5 **no ipv6 nd ra suppress** **Example:** `Router(config-if)# no ipv6 nd ra suppress`	Sending of IPv6 router advertisements is disabled by default on tunnel interfaces. This command reenables the sending of IPv6 router advertisements to allow client autoconfiguration.
Step 6 **tunnel source** {*ip-address*| *interface-type interface-number*} **Example:** `Router(config-if)# tunnel source ethernet 1/0`	Specifies the source interface type and number for the tunnel interface. **Note** The interface type and number specified in the **tunnel source** command must be configured with an IPv4 address.
Step 7 **tunnel mode ipv6ip isatap** **Example:** `Router(config-if)# tunnel mode ipv6ip isatap`	Specifies an IPv6 overlay tunnel using a ISATAP address.

Verifying IPv6 Tunnel Configuration and Operation

Perform this task to verify IPv6 tunnel configuration and operation.

SUMMARY STEPS

1. **enable**
2. **show interfaces tunnel** *number* [**accounting**]
3. **ping** [*protocol*] *destination*
4. **show ip route** [*address*[*mask*]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show interfaces tunnel** *number* [**accounting**] **Example:** `Router# show interfaces tunnel 0`	(Optional) Displays tunnel interface information. • Use the *number* argument to display information for a specified tunnel.
Step 3	**ping** [*protocol*] *destination* **Example:** `Router# ping 10.0.0.1`	(Optional) Diagnoses basic network connectivity.
Step 4	**show ip route** [*address*[*mask*]] **Example:** `Router# show ip route 10.0.0.2`	(Optional) Displays the current state of the routing table. **Note** Only the syntax relevant for this task is shown.

• Examples, page 867

Examples

Sample Output to check remote endpoint address from the ping Command

This example is a generic example suitable for both IPv6 manually configured tunnels and IPv6 over IPv4 GRE tunnels. In the example, two routers are configured to be endpoints of a tunnel. Router A has Ethernet interface 0/0 configured as tunnel interface 0 with an IPv4 address of 10.0.0.1 and an IPv6 prefix of 2001:DB8:1111:2222::1/64. Router B has Ethernet interface 0/0 configured as tunnel interface 1 with an IPv4 address of 10.0.0.2 and an IPv6 prefix of 2001:DB8:1111:2222::2/64. To verify that the tunnel source and destination addresses are configured, use the **show interfaces tunnel** command on Router A.

```
RouterA# show interfaces tunnel 0
```

```
Tunnel0 is up, line protocol is up
  Hardware is Tunnel
  MTU 1514 bytes, BW 9 Kbit, DLY 500000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation TUNNEL, loopback not set
  Keepalive not set
  Tunnel source 10.0.0.1 (Ethernet0/0), destination 10.0.0.2, fastswitch TTL 255
  Tunnel protocol/transport GRE/IP, key disabled, sequencing disabled
  Tunnel TTL 255
  Checksumming of packets disabled,  fast tunneling enabled
  Last input 00:00:14, output 00:00:04, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue :0/0 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     4 packets input, 352 bytes, 0 no buffer
     Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     8 packets output, 704 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
     0 output buffer failures, 0 output buffers swapped out
```

Sample Output from the ping Command

To check that the local endpoint is configured and working, use the **ping** command on Router A:

```
RouterA# ping 2001:DB8:1111:2222::2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:1111:2222::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 20/20/20 ms
```

Sample Output from the show ip route Command

To check that a route exists to the remote endpoint address, use the **show ip route** command:

```
RouterA# show ip route 10.0.0.2
Routing entry for 10.0.0.0/24
  Known via "connected", distance 0, metric 0 (connected, via interface)
  Routing Descriptor Blocks:
  * directly connected, via Ethernet0/0
      Route metric is 0, traffic share count is 1
```

Sample Output from the ping Command

To check that the remote endpoint address is reachable, use the **ping** command on Router A.

Note The remote endpoint address may not be reachable using the **ping** command because of filtering, but the tunnel traffic may still reach its destination.

```
RouterA# ping 10.0.0.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.0.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 20/21/28 ms
```

To check that the remote IPv6 tunnel endpoint is reachable, use the **ping** command again on Router A. The same note on filtering also applies to this example.

```
RouterA# ping 1::2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 1::2, timeout is 2 seconds:
```

```
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 20/20/20 ms
```

These steps may be repeated at the other endpoint of the tunnel.

Configuration Examples for Implementing Tunneling for IPv6

Example: Configuring Manual IPv6 Tunnels

The following example configures a manual IPv6 tunnel between Router A and Router B. In the example, tunnel interface 0 for both Router A and Router B is manually configured with a global IPv6 address. The tunnel source and destination addresses are also manually configured.

Router A Configuration

```
interface ethernet 0
 ip address 192.168.99.1 255.255.255.0
interface tunnel 0
 ipv6 address 3ffe:b00:c18:1::3/127
 tunnel source ethernet 0
 tunnel destination 192.168.30.1
 tunnel mode ipv6ip
```

Router B Configuration

```
interface ethernet 0
 ip address 192.168.30.1 255.255.255.0
interface tunnel 0
 ipv6 address 3ffe:b00:c18:1::2/127
 tunnel source ethernet 0
 tunnel destination 192.168.99.1
 tunnel mode ipv6ip
```

Example Configuring GRE Tunnels

Example: GRE Tunnel Running IS-IS and IPv6 Traffic

The following example configures a GRE tunnel running both IS-IS and IPv6 traffic between Router A and Router B:

Router A Configuration

```
ipv6 unicast-routing
clns routing
!
interface tunnel 0
 no ip address
 ipv6 address 3ffe:b00:c18:1::3/127
 ipv6 router isis
 tunnel source Ethernet 0/0
 tunnel destination 2001:DB8:1111:2222::1/64
 tunnel mode gre ipv6
!
interface Ethernet0/0
 ip address 10.0.0.1 255.255.255.0
!
router isis
 net 49.0000.0000.000a.00
```

Router B Configuration

```
ipv6 unicast-routing
clns routing
!
interface tunnel 0
 no ip address
 ipv6 address 3ffe:b00:c18:1::2/127
 ipv6 router isis
 tunnel source Ethernet 0/0
 tunnel destination 2001:DB8:1111:2222::2/64
 tunnel mode gre ipv6
!
interface Ethernet0/0
 ip address 10.0.0.2 255.255.255.0
!
router isis
 net 49.0000.0000.000b.00
 address-family ipv6
 redistribute static
 exit-address-family
```

Example: Tunnel Destination Address for IPv6 Tunnel

```
Router(config
)
# interface Tunnel0
Router(config
-if)
# no ip address
Router(config
-if)
# ipv6 router isis
Router(config
-if)
# tunnel source Ethernet 0/0
Router(config
-if)
# tunnel destination 2001:DB8:1111:2222::1/64
Router(config
-if)
# tunnel mode gre ipv6
Router(config
-if)
# exit
!
Router(config
)
# interface Ethernet0/0
```

```
Router(config
-if)
# ip address 10.0.0.1 255.255.255.0
Router(config
-if)
# exit
!
Router(config
)
# ipv6 unicast-routing
Router(config
)
# router isis

Router(config
)
# net 49.0000.0000.000a.00
```

Example: Configuring CTunnels in GRE Mode to Carry IPv6 Packets in CLNS

The following example configures a GRE CTunnel running both IS-IS and IPv6 traffic between Router A and Router B in a CLNS network. The **ctunnel mode gre** command allows tunneling between Cisco and third-party networking devices and carries both IPv4 and IPv6 traffic.

The **ctunnel mode gre** command provides a method of tunneling that is compliant with RFC 3147 and allows tunneling between Cisco equipment and third-party networking devices.

Router A

```
ipv6 unicast-routing
clns routing
interface ctunnel 102
 ipv6 address 2001:DB8:1111:2222::1/64
 ctunnel destination 49.0001.2222.2222.2222.00
 ctunnel mode gre
interface Ethernet0/1
 clns router isis
router isis
 net 49.0001.1111.1111.1111.00
```

Router B

```
ipv6 unicast-routing
clns routing
interface ctunnel 201
 ipv6 address 2001:DB8:1111:2222::2/64
 ctunnel destination 49.0001.1111.1111.1111.00
 ctunnel mode gre
interface Ethernet0/1
 clns router isis
router isis
 net 49.0001.2222.2222.2222.00
```

To turn off GRE mode and restore the CTunnel to the default Cisco encapsulation routing only between endpoints on Cisco equipment, use either the **no ctunnel mode** command or the **ctunnel mode cisco** command. The following example shows the same configuration modified to transport only IPv4 traffic.

Example: Configuring 6to4 Tunnels

The following example configures a 6to4 tunnel on a border router in an isolated IPv6 network. The IPv4 address is 192.168.99.1, which translates to the IPv6 prefix of 2002:c0a8:6301::/48. The IPv6 prefix is subnetted into 2002:c0a8:6301::/64 for the tunnel interface: 2002:c0a8:6301:1::/64 for the first IPv6

network, and 2002:c0a8:6301:2::/64 for the second IPv6 network. The static route ensures that any other traffic for the IPv6 prefix 2002::/16 is directed to tunnel interface 0 for automatic tunneling.

```
interface Ethernet0
 description IPv4 uplink
 ip address 192.168.99.1 255.255.255.0
!
interface Ethernet1
 description IPv6 local network 1
 ipv6 address 2002:c0a8:6301:1::1/64
!
interface Ethernet2
 description IPv6 local network 2
 ipv6 address 2002:c0a8:6301:2::1/64
!
interface Tunnel0
 description IPv6 uplink
 no ip address
 ipv6 address 2002:c0a8:6301::1/64
 tunnel source Ethernet 0
 tunnel mode ipv6ip 6to4
!
ipv6 route 2002::/16 tunnel 0
```

Example: Configuring IPv4-Compatible IPv6 Tunnels

The following example configures an IPv4-compatible IPv6 tunnel that allows Border Gateway Protocol (BGP) to run between a number of routers without having to configure a mesh of manual tunnels. Each router has a single IPv4-compatible tunnel, and multiple BGP sessions can run over each tunnel, one to each neighbor. Ethernet interface 0 is used as the tunnel source. The tunnel destination is automatically determined by the IPv4 address in the low-order 32 bits of an IPv4-compatible IPv6 address. Specifically, the IPv6 prefix 0:0:0:0:0:0 is concatenated to an IPv4 address (in the format 0:0:0:0:0:0:A.B.C.D or ::A.B.C.D) to create the IPv4-compatible IPv6 address. Ethernet interface 0 is configured with a global IPv6 address and an IPv4 address (the interface supports both the IPv6 and IPv4 protocol stacks).

Multiprotocol BGP is used in the example to exchange IPv6 reachability information with the peer 10.67.0.2. The IPv4 address of Ethernet interface 0 is used in the low-order 32 bits of an IPv4-compatible IPv6 address and is also used as the next-hop attribute. Using an IPv4-compatible IPv6 address for the BGP neighbor allows the IPv6 BGP session to be automatically transported over an IPv4-compatible tunnel.

```
interface tunnel 0
 tunnel source Ethernet 0
 tunnel mode ipv6ip auto-tunnel
interface ethernet 0
 ip address 10.27.0.1 255.255.255.0
 ipv6 address 3000:2222::1/64
router bgp 65000
 no synchronization
 no bgp default ipv4-unicast
neighbor ::10.67.0.2 remote-as 65002
address-family ipv6
 neighbor ::10.67.0.2 activate
 neighbor ::10.67.0.2 next-hop-self
 network 2001:2222:d00d:b10b::/64
```

Example: Configuring 6RD Tunnels

The following example shows the running configuration of a 6RD tunnel and the corresponding output of the **show tunnel 6rd** command:

```
interface Tunnel1
 ipv6 address 2001:B000:100::1/32
```

```
 tunnel source Ethernet2/1
 tunnel mode ipv6ip 6rd
 tunnel 6rd prefix 2001:B000::/32
 tunnel 6rd ipv4 prefix-len 16 suffix-len 8
end
Router# show tunnel 6rd tunnel 1
Interface Tunnel1:
  Tunnel Source: 10.1.1.1
  6RD: Operational, V6 Prefix: 2001:B000::/32
       V4 Common Prefix Length: 16, Value: 10.1.0.0
       V4 Common Suffix Length: 8, Value: 0.0.0.1
```

Example: Configuring ISATAP Tunnels

The following example shows the tunnel source defined on Ethernet 0 and the **tunnel mode** command used to configure the ISATAP tunnel. Router advertisements are enabled to allow client autoconfiguration.

```
ipv6 unicast-routing
interface tunnel 1
 tunnel source ethernet 0
 tunnel mode ipv6ip isatap
 ipv6 address 2001:DB8::/64 eui-64
 no ipv6 nd ra suppress
 exit
```

Additional References

Related Documents

Related Topic	Document Title
IPsec VTIs	Implementing IPsec in IPv6 Security
IPv6 supported feature list	"Start Here: Cisco IOS Software Release Specifics for IPv6 Features," *Cisco IOS IPv6 Configuration Guide*
CLNS tunnels	*Cisco IOS ISO CLNS Configuration Guide*
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*

Standards

Standard	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2473	*Generic Packet Tunneling in IPv6 Specification*
RFC 2893	*Transition Mechanisms for IPv6 Hosts and Routers*
RFC 3056	*Connection of IPv6 Domains via IPv4 Clouds*
RFC 4214	*Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing Tunneling for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 45 Feature Information for Implementing Tunneling for IPv6

Feature Name	Releases	Feature Information
CEFv6 Switching for 6to4 Tunnels	12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(18)SXE 12.2(12)T 12.4 15.0(1)S 15.1(1)SG	Cisco Express Forwarding switching can be used for IPv6 manually configured tunnels.
IPv6 Tunneling--6RD IPv6 Rapid Deployment	15.1(3)T	The 6RD feature allows a service provider to provide a unicast IPv6 service to customers over its IPv4 network by using encapsulation of IPv6 in IPv4.
IPv6 Tunneling--Automatic 6to4 Tunnels	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(33)SRA12.2(18)SXE 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	An automatic 6to4 tunnel allows isolated IPv6 domains to be connected over an IPv4 network to remote IPv6 networks.
IPv6 Tunneling--Automatic IPv4-Compatible Tunnels	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(18)SXE 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	Automatic IPv4-compatible tunnels use IPv4-compatible IPv6 addresses.
IPv6 Tunneling--IPv6 GRE Tunnels in CLNS Networks	12.2(25)S 12.2(28)SB 12.2(33)SRA 12.3(7)T 12.4 12.4(2)T	GRE tunneling of IPv4 and IPv6 packets through CLNS networks enables Cisco CTunnels to interoperate with networking equipment from other vendors.
IPv6 Tunneling--IP over IPv6 GRE Tunnels	12.2(30)S 12.3(7)T 12.4 12.4(2)T	GRE tunnels are links between two points, with a separate tunnel for each link.
IPv6 Tunneling--IPv4 over IPv6 Tunnels	12.2(30)S 12.2(33)SRA 12.3(7)T 12.4 12.4(2)T 15.0(1)S	IPv6 supports this feature

Feature Name	Releases	Feature Information
IPv6 Tunneling--IPv6 over IPv4 GRE Tunnels	12.0(22)S[4] 12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(17a)SX1 12.2(4)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	GRE tunnels are links between two points, with a separate tunnel for each link. The tunnels are not tied to a specific passenger or transport protocol, but in this case carry IPv6 as the passenger protocol with the GRE as the carrier protocol and IPv4 or IPv6 as the transport protocol.
IPv6 Tunneling--IPv6 over IPv6 Tunnels	12.2(30)S 12.3(7)T 12.4 12.4(2)T	IPv6 supports this feature
IPv6 Tunneling--IPv6 over UTI Using a Tunnel Line Card[5]	12.0(23)S	IPv6 supports this feature.
IPv6 Tunneling--ISATAP Tunnel Support	12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(17a)SX1 12.2(15)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S 15.1(1)SG	ISATAP is an automatic overlay tunneling mechanism that uses the underlying IPv4 network as a NBMA link layer for IPv6.
IPv6 Tunneling--Manually Configured IPv6 over IPv4 Tunnels	12.0(23)S[6] 12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	A manually configured tunnel is equivalent to a permanent link between two IPv6 domains over an IPv4 backbone.
mGRE Tunnels over IPv6	15.2(1)T	mGRE tunnels are configured to enable service providers deploy IPv6 in their core infrastructure.

4 IPv6 over IPv4 GRE tunnels are not supported on the GSR.

5 Feature is supported on the GSR only.

6 In Cisco IOS Release 12.0(23)S, the GSR provides enhanced performance for IPv6 manually configured tunnels by processing traffic on the line card.