

Implementing Network Stack IPv4 and IPv6

- Implementing Network Stack IPv4 and IPv6, on page 1
- Implementing Fallback VRF , on page 1
- Network Stack IPv4 and IPv6 Exceptions, on page 2
- IPv4 and IPv6 Functionality, on page 2
- Custom Prefix Length Selection , on page 3
- IPv6 for Cisco IOS XR Software, on page 9
- How to Implement Network Stack IPv4 and IPv6, on page 10

Implementing Network Stack IPv4 and IPv6

The Network Stack IPv4 and IPv6 features are used to configure and monitor Internet Protocol Version 4 (IPv4) and Internet Protocol Version 6 (IPv6).

Restrictions

In any Cisco IOS XR software release with IPv6 support, multiple IPv6 global addresses can be configured on an interface. However, multiple IPv6 link-local addresses on an interface are not supported.

Implementing Fallback VRF

Virtual Routing and Forwarding (VRF) is an IP technology that allows multiple instances of a routing table to coexist simultaneously on the same router. Because the routing instances are independent, the same IP addresses can be used without conflict.

If the destination prefix of a data packet does not match any route in the configured VRF, a default route is identified from the global routing table. However, using a default route needs an explicit next hop and that may not be efficient. A better option is to configure a fallback VRF route. If the destination does not have a match in the VRF table, the fallback VRF table is used. The fallback VRF can either be the global routing table or a non-global VRF table.

Restrictions

The following restrictions apply if you configure a fallback VRF route:

- You can configure only one fallback VRF route for each address family of each primary VRF.

- Ping, traceroute, or any slow path application is not supported on fallback VRF because there is no support for LPTS receive trap.
- Only 255 VRFs and 1 global table are supported ion the router.
- If you configure a static default route to a VRF, the static default route takes precedence over the fallback VRF. If you configure the default route for a VRF, the global routing table is used for a route lookup. The default route is always directed to the configured next hop.
- If a route lookup for a packet fails in the primary VRF, the packet is recycled to do route lookup in the fallback VRF. Therefore, the routing performance of the packet goes down by up to 50 percent.
- If you configure both ACL-based forwarding (ABF) VRF redirect and VRF fallback for a packet, then the packet is recycled twice. Therefore, the routing performance of the packet goes down by up to 33 percent.
- If a route for a packet is found in the fallback VRF, only the Glean IPv4 and Glean IPv6 adjacency packets are punted successfully.
- In a looped configuration, if the route for a packet is not found in both the primary and fallback VRF, the packet loops in the recycle path. Eventually, the packet is dropped in the recycle egress queue. The recycle queue is of highest priority. Therefore, if there is a high rate of looped traffic, other good recycled packets may be dropped.

Network Stack IPv4 and IPv6 Exceptions

The Network Stack feature in the Cisco IOS XR software has the following exceptions:

- In Cisco IOS XR software, the **clear ipv6 neighbors** and **show ipv6 neighbors** commands include the **location** *node-id* keyword. If a location is specified, only the neighbor entries in the specified location are displayed.
- The **ipv6 nd scavenge-timeout** command sets the lifetime for neighbor entries in the stale state. When the scavenge-timer for a neighbor entry expires, the entry is cleared.
- In Cisco IOS XR software, the **show ipv4 interface** and **show ipv6 interface** commands include the **location** *node-id* keyword. If a location is specified, only the interface entries in the specified location are displayed.
- Cisco IOS XR software allows conflicting IP address entries at the time of configuration. If an IP address conflict exists between two interfaces that are active, Cisco IOS XR software brings down the interface according to the configured conflict policy, the default policy being to bring down the higher interface instance.

  For example, if HundredGigE 0/0/0/1 conflicts with HundredGigE 0/0/0/2, then the IPv4 protocol on HundredGigE 0/0/0/2, is brought down and IPv4 remains active on HundredGigE 0/0/0/1.

IPv4 and IPv6 Functionality

When Cisco IOS XR software is configured with both an IPv4 and an IPv6 address, the interface can send and receive data on both IPv4 and IPv6 networks.

The architecture of IPv6 has been designed to allow existing IPv4 users to make the 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 Open Shortest Path First (OSPF), and multiprotocol Border Gateway Protocol (BGP).

The IPv6 neighbor discovery (nd) process uses Internet Control Message Protocol (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 keep track of neighboring routers.

Custom Prefix Length Selection

Feature Name	Release Information	Feature Description
Custom Prefix Length Selection	Release 7.4.1	By default, /48 prefix length is inserted in the LEM memory. This feature allows you to choose a custom IPv6 prefix length to be inserted into the largest exact match (LEM) memory. This feature introduces the **hw-module fib scale ipv6 custom-lem** command.

By default, /48 prefix length is inserted in the LEM memory. This feature allows you to choose a custom IPv6 prefix length to be inserted into the largest exact match (LEM) memory.

Restrictions

- Do not configure the IPv6 **internet-optimized-disable** command and the **hw-module custom-lem** command together.
- You can configure only one single length at a time. You can choose only one prefix length value to be put into the LEM memory.
- Make sure that the IPv6 length that you chose is nibble granular, that is multiples of 4.
- This feature is only supported on NCS57 line cards with no eTCAM.

Configuration Example

```
Router(config)# hw-module fib scale ipv6 custom-lem
```

Running Configuration

```
hw-module fib scale ipv6 custom-lem
```

Verification

Verify the prefix distribution with different prefix lengths.

```
Router# show dpa resources ip6route location 0/0/CPU0
Fri Jul  9 15:33:00.652 UTC

"ip6route" OFA Table (Id: 53, Scope: Global)
--------------------------------------------------
IPv6 Prefix len distribution
Prefix   Actual       Prefix   Actual
/0       1            /1       0
 /2       0            /3       0
 /4       0            /5       0
 /6       0            /7       0
 /8       0            /9       0
 /10      1            /11      0
 /12      0            /13      0
 /14      0            /15      0
 /16      3            /17      0
 /18      0            /19      0
 /20      0            /21      0
 /22      0            /23      0
 /24      0            /25      0
 /26      0            /27      0
 /28      0            /29      0
 /30      0            /31      0
 /32      0            /33      0
 /34      0            /35      0
 /36      0            /37      0
 /38      0            /39      0
 /40      0            /41      0
 /42      0            /43      0
 /44      0            /45      0
 /46      0            /47      0
 /48      0            /49      0
 /50      0            /51      0
 /52      0            /53      0
 /54      0            /55      0
 /56      0            /57      0
 /58      0            /59      0
 /60      0            /61      0
 /62      0            /63      0
 /64      0            /65      0
 /66      0            /67      0
 /68      0            /69      0
 /70      0            /71      0
 /72      0            /73      0
 /74      0            /75      0
 /76      0            /77      0
 /78      0            /79      0
 /80      0            /81      0
 /82      0            /83      0
 /84      0            /85      0
 /86      0            /87      0
 /88      0            /89      0
 /90      0            /91      0
 /92      0            /93      0
 /94      0            /95      0
 /96      0            /97      0
 /98      0            /99      0
 /100     0            /101     0
 /102     0            /103     0
 /104     1            /105     0
 /106     0            /107     0
 /108     0            /109     0
 /110     0            /111     0
 /112     0            /113     0
```

```
 /114     0            /115     0
 /116     0            /117     0
 /118     0            /119     0
 /120     0            /121     0
 /122     0            /123     0
 /124     0            /125     0
 /126     0            /127     0
 /128     1

OFA Infra Stats Summary
                 Create Requests: 7
                 Delete Requests: 0
                 Update Requests: 0
                    Get Requests: 0

                  Backwalk Stats
                 Update Requests: 0
                  Update Skipped: 0

                          Errors
               Resolve Failures: 0
                Not Found in DB: 0
                   Exists in DB: 0
                No Memory in DB: 0
              Reserve Resources: 0
              Release Resources: 0
               Update Resources: 0
                 Retry Attempts: 0
           Recovered from error: 0
              Errors from bwalk: 0

                        NPU ID: NPU-0
         Create Server API Err: 0
         Update Server API Err: 0
         Delete Server API Err: 0
```

show dpa resources ip6route location 0/0/CPU0

```
Fri Jul  9 17:42:09.728 UTC

"ip6route" OFA Table (Id: 53, Scope: Global)
-------------------------------------------------
IPv6 Prefix len distribution
Prefix   Actual       Prefix   Actual
 /0       1            /1       0
 /2       0            /3       0
 /4       0            /5       0
 /6       0            /7       0
 /8       0            /9       0
 /10      1            /11      0
 /12      0            /13      0
 /14      0            /15      0
 /16      3            /17      0
 /18      0            /19      0
 /20      0            /21      0
 /22      0            /23      0
 /24      0            /25      0
 /26      0            /27      0
 /28      0            /29      0
 /30      0            /31      0
 /32      0            /33      0
 /34      0            /35      0
 /36      0            /37      0
 /38      0            /39      0
```

```
 /40      0            /41      0
 /42      0            /43      0
 /44      0            /45      0
 /46      0            /47      0
 /48      0            /49      0
 /50      0            /51      0
 /52      0            /53      0
 /54      0            /55      0
 /56      0            /57      0
 /58      0            /59      0
 /60      0            /61      0
 /62      0            /63      0
 /64      2055         /65      0           >>>>>>>>> total prefixes with /64 length
 /66      0            /67      0
 /68      0            /69      0
 /70      0            /71      0
 /72      0            /73      0
 /74      0            /75      0
 /76      0            /77      0
 /78      0            /79      0
 /80      0            /81      0
 /82      0            /83      0
 /84      0            /85      0
 /86      0            /87      0
 /88      0            /89      0
 /90      0            /91      0
 /92      0            /93      0
 /94      0            /95      0
 /96      0            /97      0
 /98      0            /99      0
 /100     0            /101     0
 /102     0            /103     0
 /104     1            /105     0
 /106     0            /107     0
 /108     0            /109     0
 /110     0            /111     0
 /112     0            /113     0
 /114     0            /115     0
 /116     0            /117     0
 /118     0            /119     0
 /120     0            /121     0
 /122     0            /123     0
 /124     0            /125     0
 /126     0            /127     0
 /128     29

OFA Infra Stats Summary
                 Create Requests: 2090
                 Delete Requests: 0
                 Update Requests: 0
                    Get Requests: 0

                  Backwalk Stats
                 Update Requests: 0
                  Update Skipped: 0

                           Errors
               Resolve Failures: 0
                Not Found in DB: 0
                   Exists in DB: 0
                No Memory in DB: 0
              Reserve Resources: 0
              Release Resources: 0
               Update Resources: 0
```

```
              Retry Attempts: 0
           Recovered from error: 0
              Errors from bwalk: 0

                          NPU ID: NPU-0
          Create Server API Err: 0
          Update Server API Err: 0
          Delete Server API Err: 0
```

Verify the configured prefix LEM length. In the below example, the configured LEM length is indicated as 48.

```
Router# show controller fia diagshell 0 "config" location 0/0/CPU0
Fri Jul  9 15:31:45.616 UTC
    custom_feature_bfd_ipv6_protection=1
    bfd_ipv6_trap_port=208
    bcm886xx_ipv6_tunnel_enable=1
    custom_feature_li_ipv6_disable=1
    bcm886xx_ipv6_ext_hdr_enable=1
    enhanced_fib_scale_prefix_length_ipv6_long=48
    mcs_load_uc0=bfd_ipv6
    l3_vrrp_ipv6_distinct=1
    custom_feature_kbp_ipv6_uc_no_rpf_dip_sip_sharing_from_fwd_header=1
    enhanced_fib_scale_prefix_length_ipv6_short=48

    bfd_ipv6_enable=1
```

Verify the configured prefix LEM length. In the below example, the configured LEM length is indicated as 64.

```
show controller fia diagshell 0 "config" location 0/0/cpu0 | i ipv6
Fri Jul  9 17:41:39.518 UTC
    custom_feature_bfd_ipv6_protection=1
    bfd_ipv6_trap_port=208
    bcm886xx_ipv6_tunnel_enable=1
    custom_feature_li_ipv6_disable=1
    bcm886xx_ipv6_ext_hdr_enable=1
    enhanced_fib_scale_prefix_length_ipv6_long=64
    mcs_load_uc0=bfd_ipv6
    l3_vrrp_ipv6_distinct=1
    custom_feature_kbp_ipv6_uc_no_rpf_dip_sip_sharing_from_fwd_header=1
    enhanced_fib_scale_prefix_length_ipv6_short=64
    bfd_ipv6_enable=1
```

Verify the usage of resources.

```
Routers# show controllers npu resources lem loc 0/5/CPU0
Mon Jul 12 16:17:48.751 UTC
HW Resource Information
    Name                            : lem
    Asic Type                       : Jericho Plus

NPU-0
OOR Summary
        Estimated Max Entries       : 786432
        Red Threshold               : 95 %
        Yellow Threshold            : 80 %
        OOR State                   : Green


Current Usage
        Total In-Use                : 26       (0 %)
       iproute                     : 0        (0 %)
        ip6route                    : 0        (0 %)
        mplslabel                   : 1        (0 %)
```

```
        l2brmac                     : 0        (0 %)


NPU-1
OOR Summary
        Estimated Max Entries       : 786432
        Red Threshold               : 95 %
        Yellow Threshold            : 80 %
        OOR State                   : Green


Current Usage
        Total In-Use                : 26       (0 %)
        iproute                     : 0        (0 %)
        ip6route                    : 0        (0 %)
        mplslabel                   : 1        (0 %)
        l2brmac                     : 0        (0 %)


NPU-2
OOR Summary
        Estimated Max Entries       : 786432
        Red Threshold               : 95 %
        Yellow Threshold            : 80 %
        OOR State                   : Green


Current Usage
        Total In-Use                : 26       (0 %)
        iproute                     : 0        (0 %)
        ip6route                    : 0        (0 %)
        mplslabel                   : 1        (0 %)
        l2brmac                     : 0        (0 %)


NPU-3
OOR Summary
        Estimated Max Entries       : 786432
        Red Threshold               : 95 %
        Yellow Threshold            : 80 %
        OOR State                   : Green


Current Usage
        Total In-Use                : 26       (0 %)
        iproute                     : 0        (0 %)
        ip6route                    : 0        (0 %)
        mplslabel                   : 1        (0 %)
        l2brmac                     : 0        (0 %)
```

Verify the usage of resources.

```
show controllers npu resources lem location 0/0/CPU0
Fri Jul  9 17:42:34.516 UTC
HW Resource Information
    Name                            : lem
    Asic Type                       : Jericho

NPU-0
OOR Summary
        Estimated Max Entries       : 786432
        Red Threshold               : 95 %
        Yellow Threshold            : 80 %
        OOR State                   : Green
```

```
Current Usage
        Total In-Use                : 4223      (1 %)
        iproute                     : 2172      (0 %)
        ip6route                    : 2055      (0 %). >>>>>>>> LEM resources allocation
should = dpa resources for /64
        mplslabel                   : 1         (0 %)
        l2brmac                     : 0         (0 %)
```

Verify the summary of the CEF table.

```
show cef ipv6 summary
Fri Jul  9 17:51:02.788 UTC

Router ID is 192.168.1.3

IP CEF with switching (Table Version 0) for node0_RP0_CPU0

  Load balancing: L4
  Tableid 0xe0800000 (0x8a4f2748), Vrfid 0x60000000, Vrid 0x20000000, Flags 0x1019
  Vrfname default, Refcount 2179
  2090 routes, 0 protected, 0 reresolve, 0 unresolved (0 old, 0 new), 317680 bytes
    2083 rib, 0 lsd, 0 aib, 0 internal, 0 interface, 6 special, 1 default routes
    Prefix masklen distribution:
        unicast: 28 /128, 2055 /64 , 1 /10 , 1 /0  >>>>>>> cef prefixes received to LC
hardware
      multicast: 1 /128, 1 /104, 3 /16
  61 load sharing elements, 37968 bytes, 2090 references
  Shared load sharing elements with 9664 bytes, 2032 references, including:
    3 Pathlist elements, 0 recursive, 0 platform shared, 0 in retry
    3 Loadinfo elements, 0 recursive, 0 platform shared
  Exclusive load sharing elements with 28304 bytes, 58 references, including:
    58 Pathlist elements, 0 recursive, 0 platform shared, 0 in retry
    58 Loadinfo elements, 0 recursive, 0 platform shared
  0 Drop Pathlist elements
  0 route delete cache elements
  156 local route bufs received, 0 remote route bufs received,  0 mix bufs received
  2083 local routes, 0 remote routes
  2155 total local route updates processed
  0 total remote route updates processed
  0 pkts pre-routed to cust card
  0 pkts pre-routed to rp card
  0 pkts received from core card
  0 CEF route update drops, 50 revisions of existing leaves
  0 CEF route update drops due to version mis-match
  Resolution Timer: 15s
  0 prefixes modified in place
  0 deleted stale prefixes
  0 prefixes with label imposition, 0 prefixes with label information
  0 LISP EID prefixes, 0 merged, via 0 rlocs
  22 next hops
    0 incomplete next hops
  0 PD backwalks on LDIs with backup path
```

IPv6 for Cisco IOS XR 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 it became clear that the 32-bit addressing scheme of IP version 4 (IPv4) was inadequate to meet the

demands of Internet growth. After extensive discussion, it was decided to base IPng on IP but add a much larger address space and improvements such as a simplified main header and extension headers. IPv6 is described initially in RFC 2460, I*nternet Protocol, Version 6 (IPv6) Specification* issued by the Internet Engineering Task Force (IETF). Further RFCs describe the architecture and services supported by IPv6.

How to Implement Network Stack IPv4 and IPv6

This section contains the following procedures:

Configuring IPv4 Addressing

A basic and required task for configuring IP is to assign IPv4 addresses to network interfaces. Doing so enables the interfaces and allows communication with hosts on those interfaces using IPv4. An IP address identifies a location to which IP datagrams can be sent. An interface can have one primary IP address and multiple secondary addresses. Packets generated by the software always use the primary IPv4 address. Therefore, all networking devices on a segment should share the same primary network number.



Note Starting with Cisco IOS XR Release 7.10.2, the CLI no longer accepts IPv4 addresses with leading zeros. Use canonical dotted-decimal IPv4 addresses such as 198.51.100.1.

If your existing configuration, scripts, or Route Policy Language (RPL) expressions contain IPv4 addresses with leading zeros, NETCONF and YANG reject the address. This can cause configuration retrieval, synchronization, or BGP-related configuration to fail. Therefore, configure all IPv4 addresses in the correct format, ensuring that no part of the address includes leading zeros.

Associated with this task are decisions about subnetting and masking the IP addresses. A mask identifies the bits that denote the network number in an IP address. When you use the mask to subnet a network, the mask is then referred to as a *subnet mask*.



Note Cisco supports only network masks that use contiguous bits that are flush left against the network field.

Configuration Example

An IPv4 address of 192.168.1.27 and a network mask of "/8" is assigned to the **`interface HundredGigE 0/0/0/1`**.



Note The network mask can be a four-part dotted decimal address. For example, 255.0.0.0 indicates that each bit equal to 1 means the corresponding address bit belongs to the network address. The network mask can be indicated as a slash (/) and a number- a prefix length. 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). A slash must precede the decimal value, and there is no space between the IP address and the slash.

```
Router#configure HundredGigE0/0/0/1
Router(config)#interface HundredGigE 0/0/0/1
Router(config-if)#ipv4 address 192.168.1.27/8
Router(config-if)#commit
```

Running Configuration

```
Router#show running-config interface HundredGigE0/0/0/1
 ipv4 address 192.168.1.27 255.0.0.0
!
```

Verification

Verify that the HundredGigE interface is active and IPv4 is enabled.

```
Router# show ipv4 interface HundredGigE0/0/0/1

interface HundredGigE0/0/0/1 is Up, ipv4 protocol is Up
  Vrf is default (vrfid 0x60000000)
  Internet address is 192.168.1.27/8
  MTU is 1514 (1500 is available to IP)
  Helper address is not set
  Multicast reserved groups joined: 224.0.0.2 224.0.0.1
  Directed broadcast forwarding is disabled
  Outgoing access list is not set
  Inbound  access list is not set
  Proxy ARP is disabled
  ICMP redirects are never sent
  ICMP unreachables are always sent
  ICMP mask replies are never sent
  Table Id is 0xe0000000
```

Associated Commands

- ipv4 address
- show ipv4 interface

Configuring IPv6 Addressing

IPv6 addresses are configured to individual router interfaces in order to enable the forwarding of IPv6 traffic globally on the router. By default, IPv6 addresses are not configured.

Note The *ipv6-prefix* argument in the **ipv6 address** command must be in the form documented in RFC 2373 in which the address is specified in hexadecimal using 16-bit values between colons.

The **/prefix-length** argument in the **ipv6 address** command 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 must precede the decimal value.

The maximum number of characters allowed for the name of the prefix is 128, and any invalid value will be rejected.

The *ipv6-address* argument in the **ipv6 address link-local** command must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons.

IPv6 Multicast Groups

An IPv6 address must be configured on an interface for the interface to forward IPv6 traffic. Configuring a 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 address assigned to the interface
- All-nodes link-local multicast group FF02::1
- All-routers link-local multicast group FF02::2

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

Configuration Example

An IPv6 address of 2001:0DB8:0:1::1/64 is assigned to the **interface HundredGigE 0/0/0/1**:

```
Router#configure
Router(config)#interface HundredGigE 0/0/0/1
Router(config-if)#ipv6 address 2001:0DB8:0:1::1/64
Router(config-if)#commit
```

Running Configuration

```
Router#show running-config interface HundredGigE0/0/0/1

interface HundredGigE0/0/0/1
 ipv4 address 192.168.1.27 255.0.0.0
 ipv4 address 1.0.0.1 255.255.255.0 secondary
 ipv4 address 2.0.0.1 255.255.255.0 secondary
 ipv6 address 2001:db8:0:1::1/64
!
```

Verification

Verify that the HundredGigE interface is active and IPv6 is enabled.

```
Router#show ipv6 interface HundredGigE0/0/0/1
HundredGigE0/0/0/1 is Up, ipv6 protocol is Up, Vrfid is default (0x60000000)
  IPv6 is enabled, link-local address is fe80::c672:95ff:fea6:1c75
  Global unicast address(es):
    2001:db8:0:1::1, subnet is 2001:db8:0:1::/64
  Joined group address(es): ff02::1:ff00:1 ff02::1:ffa6:1c75 ff02::2
      ff02::1
  MTU is 1514 (1500 is available to IPv6)
  ICMP redirects are disabled
  ICMP unreachables are enabled
  ND DAD is enabled, number of DAD attempts 1
  ND reachable time is 0 milliseconds
  ND cache entry limit is 1000000000
  ND advertised retransmit interval is 0 milliseconds
  Hosts use stateless autoconfig for addresses.
```

```
  Outgoing access list is not set
  Inbound  access list is not set
  Table Id is 0xe0800000
  Complete protocol adjacency: 0
  Complete glean adjacency: 0
  Incomplete protocol adjacency: 0
  Incomplete glean adjacency: 0
  Dropped protocol request: 0
  Dropped glean request: 0
```

Associated Commands

- ipv6 address
- interface
- show ipv6 interface

Configuration Example

An IPv6 address of 2001:0DB8:0:1::/64 is assigned to the **interface HundredGigE 0/0/0/1**. The **eui-64** keyword configures site-local and 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.

```
Router#configure
Router(config)#interface HundredGigE 0/0/0/1
Router(config-if)#ipv6 address 2001:0DB8:0:1::/64 eui-64
Router(config-if)#commit
```

Running Configuration

```
Router#show running-config interface HundredGigE0/0/0/1

interface HundredGigE0/0/0/1
 ipv4 address 192.168.1.27 255.0.0.0
 ipv4 address 1.0.0.1 255.255.255.0 secondary
 ipv4 address 2.0.0.1 255.255.255.0 secondary
 ipv6 address 2001:db8:0:1::/64 eui-64
!
```

Verification

Verify that the HundredGigE interface is active and IPv6 is enabled.

```
Router#show ipv6 interface HundredGigE0/0/0/1
HundredGigE0/0/0/1 is Up, ipv6 protocol is Up, Vrfid is default (0x60000000)
  IPv6 is enabled, link-local address is fe80::c672:95ff:fea6:1c75
  Global unicast address(es):
    2001:db8:0:1:c672:95ff:fea6:1c75, subnet is 2001:db8:0:1::/64
  Joined group address(es): ff02::1:ffa6:1c75 ff02::2 ff02::1
  MTU is 1514 (1500 is available to IPv6)
  ICMP redirects are disabled
  ICMP unreachables are enabled
  ND DAD is enabled, number of DAD attempts 1
  ND reachable time is 0 milliseconds
  ND cache entry limit is 1000000000
  ND advertised retransmit interval is 0 milliseconds
  Hosts use stateless autoconfig for addresses.
  Outgoing access list is not set
  Inbound  access list is not set
```

```
  Table Id is 0xe0800000
  Complete protocol adjacency: 0
  Complete glean adjacency: 0
  Incomplete protocol adjacency: 0
  Incomplete glean adjacency: 0
  Dropped protocol request: 0
  Dropped glean request: 0
```

Associated Commands

- ipv6 address
- interface
- show ipv6 interface

Configuration Example

An IPv6 address of FE80::260:3EFF:FE11:6770 is assigned to the **interface HundredGigE 0/0/0/1**. The link-local keyword configures a link-local address on the interface that is used instead of the link-local address that is automatically configured when IPv6 is enabled on the interface.

```
Router#configure
Router(config)#interface HundredGigE 0/0/0/1
Router(config-if)#ipv6 address FE80::260:3EFF:FE11:6770 link-local
Router(config-if)#commit
```

Running Configuration

```
Router#show running-config interface HundredGigE0/0/0/1
interface HundredGigE0/0/0/1
 ipv6 address fe80::260:3eff:fe11:6770 link-local
!
```

Verification

Verify that the HundredGigE interface is active and IPv6 is enabled with link-local address.

```
Router#show ipv6 interface HundredGigE0/0/0/1
HundredGigE0/0/0/1 is Up, ipv6 protocol is Up, Vrfid is default (0x60000000)
  IPv6 is enabled, link-local address is fe80::260:3eff:fe11:6770
  Global unicast address(es):
    2001:db8:0:1:260:3eff:fe11:6770, subnet is 2001:db8:0:1::/64
  Joined group address(es): ff02::1:ff11:6770 ff02::2 ff02::1
  MTU is 1514 (1500 is available to IPv6)
  ICMP redirects are disabled
  ICMP unreachables are enabled
  ND DAD is enabled, number of DAD attempts 1
  ND reachable time is 0 milliseconds
  ND cache entry limit is 1000000000
  ND advertised retransmit interval is 0 milliseconds
  Hosts use stateless autoconfig for addresses.
  Outgoing access list is not set
  Inbound  access list is not set
  Table Id is 0xe0800000
  Complete protocol adjacency: 0
  Complete glean adjacency: 0
  Incomplete protocol adjacency: 0
  Incomplete glean adjacency: 0
  Dropped protocol request: 0
  Dropped glean request: 0
```

Associated Commands

- ipv6 address
- interface
- show ipv6 interface

Configuration Example

Enable IPv6 processing on the **interface HundredGigE 0/0/0/1**; that has not been configured with an explicit IPv6 address.

```
Router#configure
Router(config)#interface HundredGigE 0/0/0/1
Router(config-if)#ipv6 enable
Router(config-if)#commit
```

Running Configuration

```
Router#show running-config interface HundredGigE0/0/0/1
interface HundredGigE0/0/0/1

ipv6 enable
!
```

Verification

Verify that the HundredGigE interface is active and IPv6 is enabled.

```
Router#show ipv6 interface HundredGigE0/0/0/1
HundredGigE0/0/0/1 is Up, ipv6 protocol is Up, Vrfid is default (0x60000000)
  IPv6 is enabled, link-local address is fe80::c672:95ff:fea6:1c75
  No global unicast address is configured
  Joined group address(es): ff02::1:ffa6:1c75 ff02::2 ff02::1
  MTU is 1514 (1500 is available to IPv6)
  ICMP redirects are disabled
  ICMP unreachables are enabled
  ND DAD is enabled, number of DAD attempts 1
  ND reachable time is 0 milliseconds
  ND cache entry limit is 1000000000
  ND advertised retransmit interval is 0 milliseconds
  Hosts use stateless autoconfig for addresses.
  Outgoing access list is not set
  Inbound  access list is not set
  Table Id is 0xe0800000
  Complete protocol adjacency: 0
  Complete glean adjacency: 0
  Incomplete protocol adjacency: 0
  Incomplete glean adjacency: 0
  Dropped protocol request: 0
  Dropped glean request: 0
```

Associated Commands

- ipv6 enable
- interface
- show ipv6 interface

Configure Fallback VRF

You can configure a fallback VRF for a destination route that does not match any routes in the configured VRF.

The following example shows how to configure the **fallback-vrf** command for a destination that does not match any routes in the configured VRF.

```
Router# configure
Router(config)# vrf vrf1
Router(config-vrf)# fallback-vrf vrf2
```

Verification

To verify the fallback VRF details, use the **show cef vrf** *vrf-name ipv4-prefix/ipv6-prefix* **hardware egress location** *line-card-location* command:

```
Router# show cef vrf vrf100 192.0.2.1 hardware egress location 0/1/CPU0
0.0.0.0/0, version 0, proxy default, internal 0x1200011 0x0 (ptr 0x8983f534) [1], 0x0
(0x894fa728), 0x0 (0x0)
Updated Mar 21 14:01:43.765
Prefix Len 0, traffic index 0, precedence n/a, priority 15
   via 0.0.0.0/32, 0 dependencies, weight 0, class 0 [flags 0x0]
    path-idx 0 NHID 0x0 [0x8871b168 0x0]
    next hop VRF - 'vrf200', table - 0xe0000008
    next hop 0.0.0.0/32
LEAF - HAL pd context :
sub-type : IPV4, ecd_marked:0, has_collapsed_ldi:0
collapse_bwalk_required:0, ecdv2_marked:0,
HW Walk:
```

Assigning Multiple IP Addresses to Network Interfaces

The Cisco IOS XR software supports multiple IP addresses (secondary addresses) per interface. You can specify an unlimited number of secondary addresses. Secondary IP addresses can be used in a variety of situations. The following are the most common applications:

- There might not be enough host addresses for a particular network segment. For example, suppose your subnetting allows up to 254 hosts per logical subnet, but on one physical subnet you must have 300 host addresses. Using secondary IP addresses on the routers or access servers allows you to have two logical subnets using one physical subnet.
- Many older networks were built using Level 2 bridges, and were not subnetted. The judicious use of secondary addresses can aid in the transition to a subnetted, router-based network. Routers on an older, bridged segment can easily be made aware that many subnets are on that segment.
- Two subnets of a single network might otherwise be separated by another network. You can create a single network from subnets that are physically separated by another network by using a secondary address. In these instances, the first network is *extended*, or layered on top of the second network. Note that a subnet cannot appear on more than one active interface of the router at a time.

Note If any router on a network segment uses a secondary IPv4 address, all other routers on that same segment must also use a secondary address from the same network or subnet.

Caution Inconsistent use of secondary addresses on a network segment can quickly cause routing loops.

Configuration Example

A secondary IPv4 address of 192.168.1.27 is assigned to the Hundredgige interface-0/0/0/1.

Note: For IPv6, an interface can have multiple IPv6 addresses without specifying the **secondary** keyword.

```
Router# configure
Router(config)# interface HundredGigE 0/0/0/1
Router(config-if)# ipv4 address 192.168.1.27 255.255.255.0 secondary
Router(config-if)#commit
```

Running Configuration

```
Router#show running-config interface HundredGigE0/0/0/1
interface HundredGigE0/0/0/1
 ipv4 address 192.168.1.27 255.255.255.0 secondary
!
```

Verification

```
Router#show ipv4 interface HundredGigE0/0/0/1
HundredGigE0/0/0/1 is Up, ipv4 protocol is Up
  Vrf is default (vrfid 0x60000000)
  Internet address is unassigned
  Secondary address 192.168.1.27/24
  MTU is 1514 (1500 is available to IP)
  Helper address is not set
  Multicast reserved groups joined: 224.0.0.2 224.0.0.1
  Directed broadcast forwarding is disabled
  Outgoing access list is not set
  Inbound  access list is not set
  Proxy ARP is disabled
  ICMP redirects are never sent
  ICMP unreachables are always sent
  ICMP mask replies are never sent
  Table Id is 0xe0000000
```

Associated Commands

- ipv4 address
- show ipv4 interface

Configuring IPv4 and IPv6 Protocol Stacks

This task configures an interface in a Cisco networking device to support both 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—the interface can send and receive data on both IPv4 and IPv6 networks.

Configuration Example

An IPv4 address of 192.168.99.1 and an IPv6 address of 2001:0DB8:c18:1::3/64 is configured on the **interface HundredGigE 0/0/0/1**.

```
Router#configure
Router(config)#interface HundredGigE 0/0/0/1
Router(config-if)#ipv4 address 192.168.99.1 255.255.255.0
Router(config-if)#ipv6 address 2001:0DB8:c18:1::3/64
Router(config-if)#commit
```

Running Configuration

```
Router# show running-config interface HundredGigE0/0/0/1
 ipv4 address 192.168.99.1 255.255.255.0
 ipv6 address 2001:db8:c18:1::3/64
!
```

Verification

Verify that the HundredGigE interface is active and IPv4 and IPv6 are enabled.

```
Router#show ipv4 interface HundredGigE0/0/0/1
HundredGigE0/0/0/1 is Up, ipv4 protocol is Up
  Vrf is default (vrfid 0x60000000)
  Internet address is 192.168.99.1/24
  MTU is 1514 (1500 is available to IP)
  Helper address is not set
  Multicast reserved groups joined: 224.0.0.2 224.0.0.1
  Directed broadcast forwarding is disabled
  Outgoing access list is not set
  Inbound  access list is not set
  Proxy ARP is disabled
  ICMP redirects are never sent
  ICMP unreachables are always sent
  ICMP mask replies are never sent
  Table Id is 0xe0000000

Router#show ipv6 interface HundredGigE0/0/0/1
HundredGigE0/0/0/1 is Up, ipv6 protocol is Up, Vrfid is default (0x60000000)
  IPv6 is enabled, link-local address is fe80::c672:95ff:fea6:1c75
  Global unicast address(es):
    2001:db8:c18:1::3, subnet is 2001:db8:c18:1::/64
  Joined group address(es): ff02::1:ff00:3 ff02::1:ffa6:1c75 ff02::2
      ff02::1
  MTU is 1514 (1500 is available to IPv6)
  ICMP redirects are disabled
  ICMP unreachables are enabled
  ND DAD is enabled, number of DAD attempts 1
  ND reachable time is 0 milliseconds
  ND cache entry limit is 1000000000
  ND advertised retransmit interval is 0 milliseconds
```

```
Hosts use stateless autoconfig for addresses.
Outgoing access list is not set
Inbound  access list is not set
Table Id is 0xe0800000
Complete protocol adjacency: 0
Complete glean adjacency: 0
Incomplete protocol adjacency: 0
Incomplete glean adjacency: 0
Dropped protocol request: 0
Dropped glean request: 0
```

Associated Commands

- ipv4 address
- ipv6 address
- show ipv4 interface
- show ipv6 interface

Enabling IPv4 Processing on an Unnumbered Interface

This section describes the process of enabling an IPv4 point-to-point interface without assigning an explicit IP address to the interface. Whenever the unnumbered interface generates a packet (for example, for a routing update), it uses the address of the interface you specified as the source address of the IP packet. It also uses the specified interface address in determining which routing processes are sending updates over the unnumbered interface. Restrictions are as follows:

- Interfaces using High-Level Data Link Control (HDLC), PPP, and Frame Relay encapsulations can be unnumbered. Serial interfaces using Frame Relay encapsulation can also be unnumbered, but the interface must be a point-to-point sub-interface.
- You cannot use the **ping** EXEC command to determine whether the interface is up, because the interface has no IP address. The Simple Network Management Protocol (SNMP) can be used to remotely monitor interface status.
- You cannot support IP security options on an unnumbered interface.

If you are configuring Intermediate System-to-Intermediate System (IS-IS) across a serial line, you should configure the serial interfaces as unnumbered, which allows you to conform with RFC 1195, which states that IP addresses are not required on each interface.

Configuration Example

Enables an IPv4 point-to-point interface without assigning an explicit IP address to the interface.

```
Router#configure
Router(config)#interface HundredGigE 0/0/0/1
Router(config-if)#ipv4 unnumbered loopback 0
Router(config-if)#commit
```

Running Configuration

```
Router#show running-config interface HundredGigE0/0/0/1
interface HundredGigE0/0/0/1
 ipv4 point-to-point
```

```
 ipv4 unnumbered Loopback0
!
```

Verification

```
Router#show interface HundredGigE0/0/0/1
HundredGigE0/0/0/1 is up, line protocol is up
  Interface state transitions: 5
  Hardware is Hundredgige, address is 00e2.2a33.445b (bia 00e2.2a33.445b)
  Layer 1 Transport Mode is LAN
  Internet address is 10.0.0.2/32
  MTU 1514 bytes, BW 10000000 Kbit (Max: 10000000 Kbit)
     reliability 255/255, txload 194/255, rxload 0/255
  Encapsulation ARPA,
  Full-duplex, 10000Mb/s, link type is force-up
  output flow control is off, input flow control is off
  Carrier delay (up) is 10 msec
  loopback not set,
  Last link flapped 01:38:49
  ARP type ARPA, ARP timeout 04:00:00
  Last input 00:00:00, output 00:00:00
  Last clearing of "show interface" counters 02:34:16
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 7647051000 bits/sec, 12254894 packets/sec
     1061401410 packets input, 82789675614 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 5 broadcast packets, 19429 multicast packets
              0 runts, 0 giants, 0 throttles, 0 parity
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     76895885948 packets output, 6192569128048 bytes, 0 total output drops
     Output 7 broadcast packets, 18916 multicast packets
     0 output errors, 0 underruns, 0 applique, 0 resets
     0 output buffer failures, 0 output buffers swapped out
     2 carrier transitions

Router #show run int lo 0
interface Loopback0
 ipv4 address 10.0.0.2 255.255.255.255
```

Associated Commands

- ipv4 unnumbered
- show interfaces

IPv4 ICMP Rate Limiting

The IPv4 ICMP rate limiting feature limits the rate that IPv4 ICMP destination unreachable messages are generated. The Cisco IOS XR software maintains two timers: one for general destination unreachable messages and one for DF destination unreachable messages. Both share the same time limits and defaults. If the DF keyword is not configured, the icmp ipv4 rate-limit unreachable command sets the time values for DF destination unreachable messages. If the DF keyword is configured, its time values remain independent from those of general destination unreachable messages.

Configuration Example

Limits the rate that IPv4 ICMP destination unreachable messages are generated every 1000 millisecond.

The **DF** keyword, which is optional limits the rate at which ICMP destination unreachable messages are sent when code 4 fragmentation is needed and Don't Fragment (DF) is set, as specified in the IP header of the ICMP destination unreachable message.

```
Router#configure
Router(config)#icmp ipv4 rate-limit unreachable 1000
Router(config)#icmp ipv4 rate-limit unreachable DF 1000
Router(config)#commit
```

Running Configuration

```
Router#show running-config | in icmp
Building configuration...
icmp ipv4 rate-limit unreachable DF 1000
icmp ipv4 rate-limit unreachable 1000
```

Verification

```
Router#show ipv4 interface HundredGigE0/0/0/2
HundredGigE0/0/0/2 is Up, ipv4 protocol is Up
  Vrf is default (vrfid 0x60000000)
  Internet address is 192.85.1.1/24
  MTU is 1514 (1500 is available to IP)
  Helper address is not set
  Multicast reserved groups joined: 224.0.0.2 224.0.0.1 224.0.0.2
      224.0.0.5 224.0.0.6
  Directed broadcast forwarding is disabled
  Outgoing access list is not set
  Inbound  common access list is not set, access list is not set
  Proxy ARP is disabled
  ICMP redirects are never sent
  ICMP unreachables are always sent
  ICMP mask replies are never sent
  Table Id is 0xe0000000
```

The number of ICMP unreachable messages that were we sent or received can be identified using the **show ipv4 traffic** command.

```
Router# show ipv4 traffic
ICMP statistics:
  Sent: 0 admin unreachable, 5 network unreachable
        0 host unreachable, 0 protocol unreachable
        0 port unreachable, 0 fragment unreachable
        0 time to live exceeded, 0 reassembly ttl exceeded
        0 echo request, 0 echo reply
        0 mask request, 0 mask reply
        0 parameter error, 0 redirects
        5 total
  Rcvd: 0 admin unreachable, 0 network unreachable
        0 host unreachable, 0 protocol unreachable
        0 port unreachable, 0 fragment unreachable
        0 time to live exceeded, 0 reassembly ttl exceeded
        0 echo request, 0 echo reply
        0 mask request, 0 mask reply
        0 redirect, 0 parameter error
        0 source quench, 0 timestamp, 0 timestamp reply
        0 router advertisement, 0 router solicitation
        0 total, 0 checksum errors, 0 unknown
```

Associated Commands

- icmp ipv4 rate-limit unreachable

- show ipv4 traffic

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, IPv6 ICMP error messages are not 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.

Configuration Example

Configure the interval for 50 milliseconds and the bucket size for 20 tokens, for IPv6 ICMP error messages.

- The milliseconds argument specifies the interval between tokens being added to the bucket.
- The optional bucketsize argument defines the maximum number of tokens stored in the bucket.

```
Router#configure
Router(config)#ipv6 icmp error-interval 50 20
Router(config)#commit
```

Running Configuration

```
Router#show running-config
Building configuration...
!! IOS XR Configuration version = 6.0.0.26I
!! Last configuration change at Mon Dec 14 22:07:35 2015 by root
!
hostname test-83
logging console debugging
username root
 group root-lr
 group cisco-support
 secret 5 $1$d2NC$RbAdqdU7kw/kEJoMP/IJG1
!
cdp
ipv6 icmp error-interval 50 20
icmp ipv4 rate-limit unreachable DF 1000
icmp ipv4 rate-limit unreachable 1000
ipv4 conflict-policy static
```

Associated Commands

- ipv6 icmp error-interval

Selecting Flexible Source IP

You can select flexible source IP address in the Internet Control Message Protocol (ICMP) response packet to respond to a failure.

Configuration Example

Enables RFC compliance for source address selection.

```
Router#configure
Router(config)#icmp ipv4 source rfc
Router(config)#commit
```

Running Configuration

```
Router#show running-config | in source rfc
Building configuration...
icmp ipv4 source rfc
```

Associated Commands

Configuring IPARM Conflict Resolution

This task sets the IP Address Repository Manager (IPARM) address conflict resolution parameters:

- Static Policy Resolution
- Longest Prefix Address Conflict Resolution
- Highest IP Address Conflict Resolution
- Route-Tag Support for Connected Routes

Static Policy Resolution

The static policy resolution configuration prevents new address configurations from affecting interfaces that are currently running.

Configuration Example

Sets the conflict policy to static, that is, prevents new interface addresses from affecting the currently running interface.

```
Router#configure
Router(config)#ipv4 conflict-policy static
*/For IPv6, use the ipv6 conflict-policy static command/*
Router(config)#commit
```

Running Configuration

```
Router#show running-config | in ipv4 config
Building configuration...
!! IOS XR Configuration version = 6.0.0.26I
!! Last configuration change at Mon Dec 14 21:57:27 2015 by root
!
hostname sample-83
logging console debugging
```

```
username root
 group root-lr
 group test
 secret 5 $1$d2NC$RbAdqdU7kw/eKJpMo/GJI1
!
cdp
ipv4 conflict-policy static
interface Loopback0
 ipv4 address 1.1.1.1 255.255.255.255
!
…..
```

Verification

```
Router#show arm ipv4 conflicts
F Forced down
| Down interface & addr             Up interface & addr VRF

F Te0/0/0/19 192.85.1.2/24   HundredGigE0/0/0/1  192.85.1.1/24 default

Forced down interface         Up interface                  VRF
```

Associated Commands

- ipv4 conflict-policy
- ipv6 conflict-policy

Longest Prefix Address Conflict Resolution

This conflict resolution policy attempts to give highest precedence to the IP address that has the longest prefix length, that is, all addresses within the conflict-set that do not conflict with the longest prefix address of the currently running interface are allowed to run as well.

Configuration Example

Configures longest prefix address conflict resolution.

```
Router# configure
Router(config)# ipv4 conflict-policy longest-prefix
*/For IPv6, use the ipv6 conflict-policy command*/
Router(config)# commit
```

Running Configuration

```
Router# show running-config | in longest-prefix
Building configuration...
ipv4 conflict-policy longest-prefix
```

Verification

```
Router#show arm ipv4 conflicts
F Forced down
| Down interface & addr             Up interface & addr VRF

F Te0/0/0/19 192.85.1.2/24   HundredGigE0/0/0/1  192.85.1.1/24 default

Forced down interface         Up interface                  VRF
```

Highest IP Address Conflict Resolution

This conflict resolution policy attempts to give highest precedence to the IP address that has the highest value, that is, the IP address with the highest value gets precedence.

Configuration

Configures highest IP address conflict resolution.

```
Router# configure
Router(config)#ipv4 conflict-policy highest-ip
*/For IPv6, use the ipv6 conflict-policy highest-ip command/*
Router(config)#commit
```

Running Configuration

```
Router#show running-config | in highest-ip
Building configuration...
ipv4 conflict-policy highest-ip
```

Verification

```
Router#show arm ipv4 conflicts
F Forced down
| Down interface & addr             Up interface & addr VRF

F Te0/0/0/19 192.85.1.2/24   HundredGigE0/0/0/1  192.85.1.1/24 default

Forced down interface         Up interface                   VRF
```

Route-Tag Support for Connected Routes

The Route-Tag Support for Connected Routes feature attaches a tag with all IPv4 and IPv6 addresses of an interface. The tag is propagated from the IPv4 and IPv6 management agents (MA) to the IPv4 and IPv6 address repository managers (ARM) to routing protocols, thus enabling the user to control the redistribution of connected routes by looking at the route tags, by using routing policy language (RPL) scripts. This prevents the redistribution of some interfaces, by checking for route tags in a route policy. The route tag feature is already available for static routes and connected routes (interfaces) wherein the route tags are matched to policies and redistribution can be prevented.

Configuration Example

Specifies an IPv4 address 10.0.54.2/30 that has a route tag of 20 to the **interface HundredGigE 0/0/0/1**.

```
Router#configure
Router(config)#interface HundredGigE 0/0/0/1
Router(config-if)#ipv4 address 10.0.54.2/30 route-tag 1899
Router(config)#commit
```

Running Configuration

```
Router#show running-config interface HundredGigE0/0/0/1
interface HundredGigE0/0/0/1
 ipv4 address 10.0.54.2/30 route-tag 1899
!
```

Verification

Verify the parameters of the route.

```
Router#show route 10.0.54.2
Routing entry for 10.0.54.2/32
  Known via "local", distance 0, metric 0 (connected)
  Tag 1899
Routing Descriptor Blocks
    directly connected, via HundredGigE0/0/0/1
      Route metric is 0
  No advertising protos.
```

Associated Commands

- route-tag

Larger IPv6 Address Space

The primary motivation for IPv6 is the need to meet the anticipated future demand for globally unique IP addresses. Applications such as mobile Internet-enabled devices (such as personal digital assistants [PDAs], telephones, and cars), home-area networks (HANs), and wireless data services are driving 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 and the use of Network Address Translation (NAT); therefore, IPv6 enables new application protocols that do not require special processing by border routers 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:

2001:0DB8:7654:3210:FEDC:BA98:7654:3210

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

It is common for IPv6 addresses to contain successive hexadecimal fields of zeros. To make IPv6 addresses less cumbersome, two colons (::) can 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.) Table 1: Compressed IPv6 Address Formats, on page 27 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 1: Compressed IPv6 Address Formats

IPv6 Address Type	Preferred Format	Compressed Format
Unicast	2001:0:0:0:0DB8:800:200C:417A	1080::0DB8:800:200C: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 Table 1: Compressed IPv6 Address Formats, on page 27 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 routers do not forward packets that have the IPv6 loopback address as their source or destination address.

The unspecified address listed in Table 1: Compressed IPv6 Address Formats, on page 27 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* argument must be in the form documented in RFC 2373, in which the address is specified in hexadecimal using 16-bit values between colons. The prefix length is a decimal value that indicates how many of the high-order contiguous bits of the address compose the prefix (the network portion of the address). For example, 2001:0DB8:8086:6502::/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 IOS XR software supports the following IPv6 unicast address types:

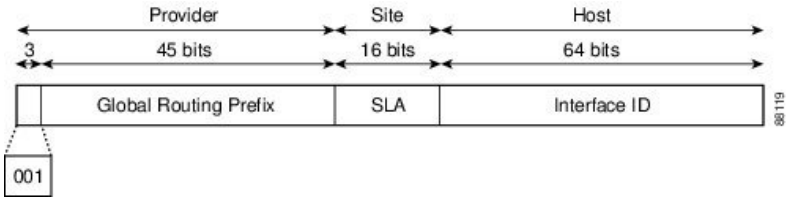
- Global aggregatable address
- Site-local address (proposal to remove by IETF)
- Link-local address
- IPv4-compatible IPv6 address

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 Internet service providers (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 current global unicast address allocation uses the range of addresses that start with binary value 001 (2000::/3). This 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 IETF decided to remove the TLS 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 is 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 interfaces and FDDI interfaces), the first three octets (24 bits) are taken from the Organizationally Unique Identifier (OUI) of the 48-bit link-layer address (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 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 router is used to construct the identifier for interfaces using PPP.

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

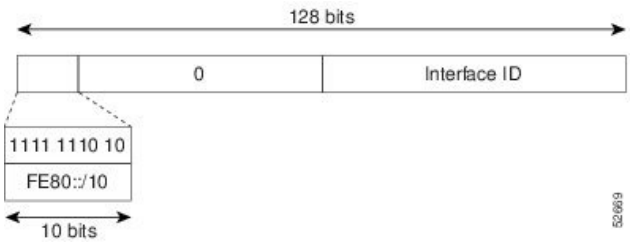
1. The router is queried for MAC addresses (from the pool of MAC addresses in the router).
2. If no MAC address is available, the serial number of the Route Processor (RP) or line card (LC) is used to form the link-local address.

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 site-local or globally unique addresses to communicate. This figure below shows the structure of a link-local address.

IPv6 routers must not forward packets that have link-local source or destination addresses to other links.

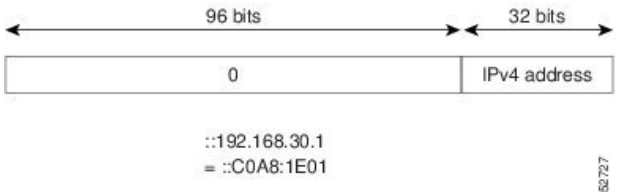
Figure 2: Link-Local Address Format



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. This figure below shows the structure of an IPv4-compatible IPv6 address and a few acceptable formats for the address.

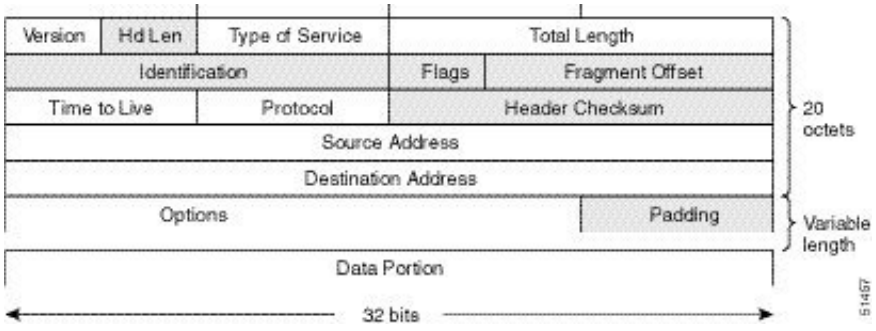
Figure 3: IPv4-Compatible IPv6 Address Format



Simplified IPv6 Packet Header

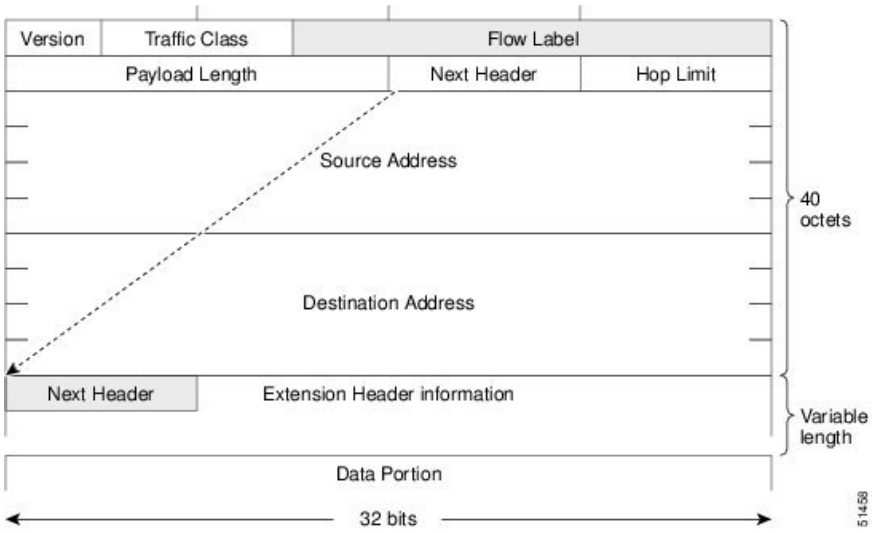
The basic IPv4 packet header has 12 fields with a total size of 20 octets (160 bits). The 12 fields may be followed by an Options field, which is followed by a data portion that is usually the transport-layer packet. The variable length of the Options field adds to the total size of the IPv4 packet header.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

Figure 4: IPv4 Packet Header Format



The basic IPv6 packet header has 8 fields with a total size of 40 octets (320 bits). Fields were removed from the IPv6 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. (In IPv4, the User Datagram Protocol (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 5: IPv6 Packet Header Format



This table lists the fields in the basic IPv6 packet header.

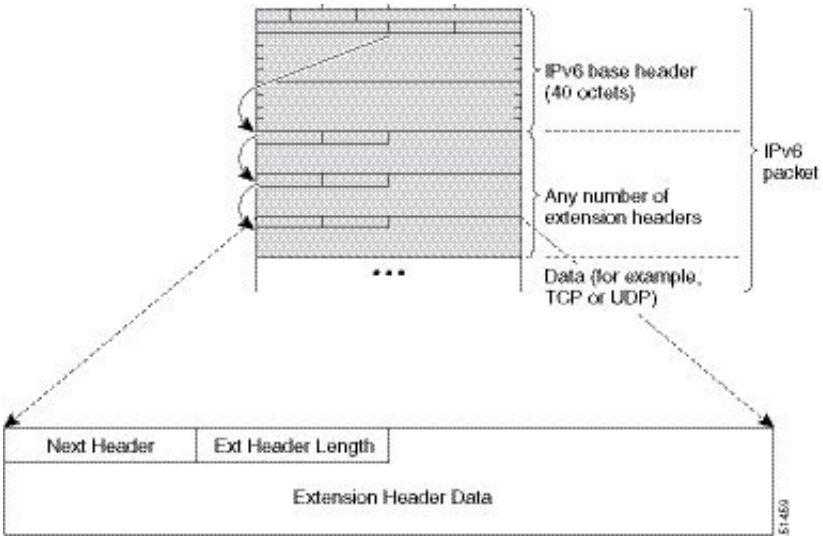
Table 2: Basic IPv6 Packet Header Fields

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.
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.
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 routers that an IPv6 packet can pass through before the packet is considered invalid. Each router decrements the value by one. Because no checksum is in the IPv6 header, the router 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.

Field	Description
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. Together, 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. This figure below shows the IPv6 extension header format.

Figure 6: IPv6 Extension Header Format



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

Table 3: 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.

Header Type	Next Header Value	Description
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 header	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 header	To be done by IANA	Extension headers used by mobile nodes, correspondent nodes, and home agents in all messaging related to the creation and management of bindings.

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.

In IPv4, the minimum link MTU is 68 octets, which means that the MTU size of every link along a given data path must support an MTU size of at least 68 octets. In IPv6, the minimum link MTU is 1280 octets. We recommend using an MTU value of 1500 octets for IPv6 links.

Note Path MTU discovery is supported only for applications using TCP.

IPv6 Neighbor Discovery

The IPv6 neighbor discovery (ND) 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 keep track of neighboring routers.

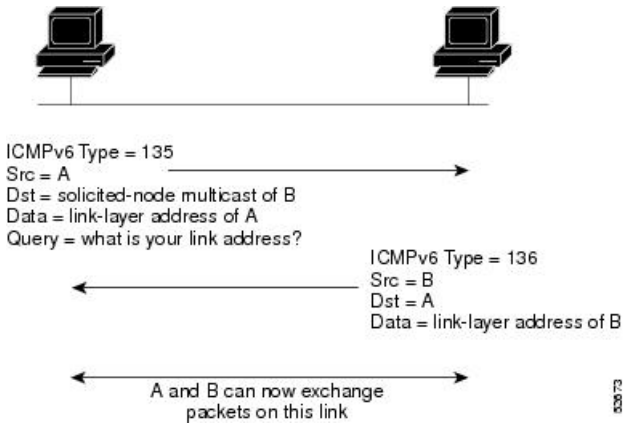
As all incoming control traffic goes through LPTS policer, if the ND packets come in a burst they are policed according to the configuration. For more details on LPTS, see LPTS Overview.

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. 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 7: 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 verifying 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 routers). 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 that a neighbor advertisement message in response to a neighbor solicitation message has been received. 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 router 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.

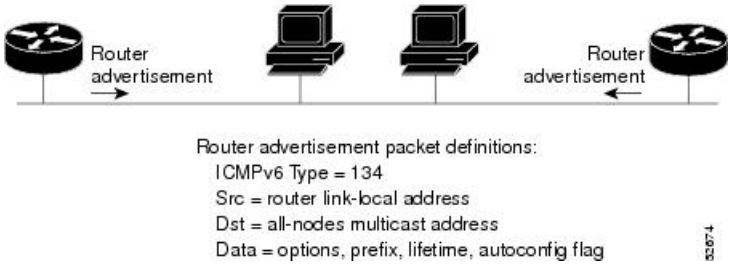
Neighbor solicitation messages are also used in the stateless autoconfiguration process to verify the uniqueness of unicast IPv6 addresses before the addresses are assigned to an interface. Duplicate address detection is performed first on a new, link-local IPv6 address before the address is assigned 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 checked 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 IOS XR software does not check the uniqueness of anycast or global addresses that are generated from 64-bit interface identifiers.

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 router. The router advertisement messages are sent to the all-nodes multicast address.

Figure 8: IPv6 Neighbor Discovery—Router Advertisement Message



Router advertisement 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 statefull) that can be completed
- Default router information (whether the router sending the advertisement should be used as a default router and, if so, the amount of time, in seconds, that the router should be used as a default router)
- Additional information for hosts, such as the hop limit and MTU a host should use in packets that it originates

Router advertisements are also sent in response to router solicitation messages. Router 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 router advertisement message. Given that router solicitation messages are usually sent by hosts at system startup (the host does not have a configured unicast address), the source address in router 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 router solicitation message is used as the source address in the message. The destination address in router solicitation messages is the all-routers multicast address with a scope of the link. When a router advertisement is sent in response to a router solicitation, the destination address in the router advertisement message is the unicast address of the source of the router solicitation message.

The following router advertisement message parameters can be configured:

- The time interval between periodic router advertisement messages
- The “router lifetime” value, which indicates the usefulness of a router as the default router (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 router advertisement messages (with default values) is automatically enabled on Ethernet and FDDI interfaces. For other interface types, the sending of router advertisement messages must be manually configured by using the **no ipv6 nd suppress-ra** command in interface configuration mode. The sending of router advertisement messages can be disabled on individual interfaces by using the **ipv6 nd suppress-ra** command in interface configuration mode.

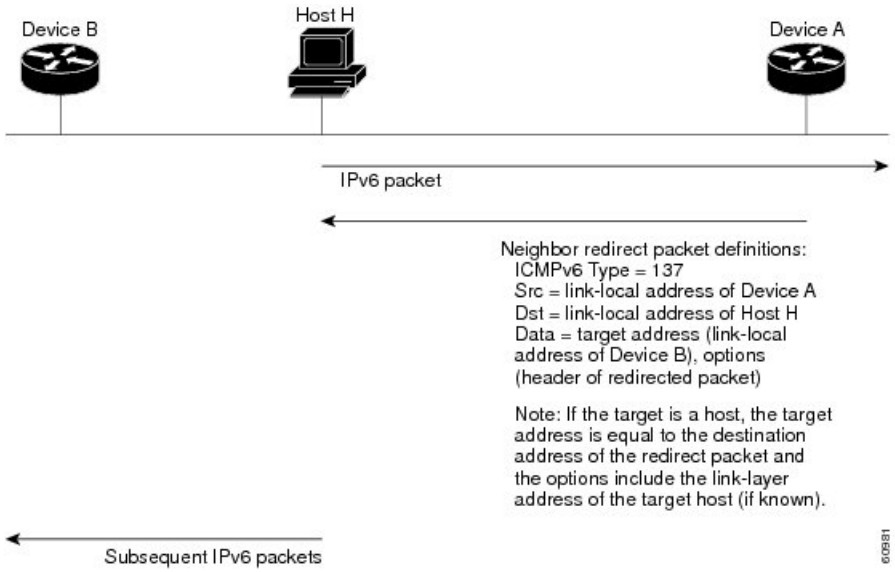


Note For stateless autoconfiguration to work properly, the advertised prefix length in router advertisement messages must always be 64 bits.

IPv6 Neighbor Redirect Message

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.

Figure 9: IPv6 Neighbor Discovery—Neighbor Redirect Message



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

After forwarding a packet, a router 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 router.
- The packet is about to be sent out the interface on which it was received.
- The router 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** global configuration command to limit the rate at which the router generates all IPv6 ICMP error messages, including neighbor redirect messages, which ultimately reduces link-layer congestion.

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

IPv6 Neighbor Discovery Proxy

Table 4: Feature History Table

Feature Name	Release Information	Feature Description
IPv6 Neighbor Discovery Proxy	Release 7.6.1	Using Internet Control Message Protocol version 6 (ICMPv6) messages, IPv6 hosts in the same subnet prefix but separated by a Layer 2 broadcast domain can now communicate with each other. The functionality allows for this communication without an additional gateway or prefix. It ensures more secure communication between the hosts on the same subnet. This feature introduces the following command: ipv6 nd proxy-nd

In IPv6 networks, the Neighbor Discovery Protocol (NDP) uses ICMPv6 messages and solicited-node multicast addresses to track and discover the other IPv6 hosts present on the other side of connected interfaces. As part of this process, a host queries for other node link-layer addresses to verify neighbor reachability using Neighbor Solicitation (NS) messages. In response to the NS messages, ND checks and sends a Neighbor Advertisement (NA) information to neighbors with the MAC address of the proxy interface.

Configure IPv6 ND Proxy

This is an example to configure ND proxy on an interface.

```
Router#configure terminal
Router(config)#interface HundredGigE0/5/0/11
Router(config-if)#ipv6 nd proxy-nd
Router(config-if)#commit
```

Note

- This feature is not supported on Cisco NCS 5700 Series routers and routers with the Cisco NC57 line cards installed and that are operating in native or compatibility mode.
- To disable this feature, use **no ipv6 nd proxy-nd** command. Once you disable this feature, the interface no longer proxy for any requests, which are not targeted for the IPv6 addresses that are configured on it.

Address Repository Manager

IPv4 and IPv6 Address Repository Manager (IPARM) enforces the uniqueness of global IP addresses configured in the system, and provides global IP address information dissemination to processes on route processors

(RPs) and line cards (LCs) using the IP address consumer application program interfaces (APIs), which includes unnumbered interface information.

Address Conflict Resolution

There are two parts to conflict resolution; the conflict database and the conflict set definition.

Conflict Database

IPARM maintains a global conflict database. IP addresses that conflict with each other are maintained in lists called conflict sets. These conflict sets make up the global conflict database.

A set of IP addresses are said to be part of a conflict set if at least one prefix in the set conflicts with every other IP address belonging to the same set. For example, the following four addresses are part of a single conflict set.

address 1: 10.1.1.1/16

address 2: 10.2.1.1/16

address 3: 10.3.1.1/16

address 4: 10.4.1.1/8

When a conflicting IP address is added to a conflict set, an algorithm runs through the set to determine the highest precedence address within the set.

This conflict policy algorithm is deterministic, that is, the user can tell which addresses on the interface are enabled or disabled. The address on the interface that is enabled is declared as the highest precedence ip address for that conflict set.

The conflict policy algorithm determines the highest precedence ip address within the set.