Implementing Multicast

Implementing Layer-3 Multicast Routing

Multicast routing allows a host to send packets to a subset of all hosts as a group transmission rather than to a single host, as in unicast transmission, or to all hosts, as in broadcast transmission. The subset of hosts is known as group members and are identified by a single multicast group address that falls under the IP Class D address range from 224.0.0.0 through 239.255.255.255.

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.

The following protocols are supported to implement multicast routing:

- IGMP—IGMP is used between hosts on a network (for example, LAN) and the routers on that network to track the multicast groups of which hosts are members.
- PIM SSM— Protocol Independent Multicast in Source-Specific Multicast (PIM-SSM) has the ability to report interest in receiving packets from specific source addresses (or from all but the specific source addresses), to an IP multicast address.

Note MLD Snooping is not supported until Cisco IOS XR Release 6.5.3.

Prerequisites for Implementing Multicast Routing

- You must install and activate the multicast RPM package.
- You must be familiar with IPv4 multicast routing configuration tasks and concepts.
- Unicast routing must be operational.

Restrictions

- Multicast over access or core pseudowire is not supported.
- NSF is enabled by default. You can't configure **nsr** under multicast-routing manually.

Enabling Multicast

Configuration Example

Enables multicast routing and forwarding on all new and existing interfaces.

```
Router#config
Router(config)#multicast-routing
Router(config-mcast)#address-family ipv4
Router(config-mcast-default-ipv4)#interface all enable
*/In the above command, you can also indicate a specific interface (For example, interface
 TenGigE0/0/0/3)
for enabling multicast only on that interface/*
Router(config-mcast-default-ipv4)#commit
```

Running Configuration

```
Router#show running multicast routing
multicast-routing
 address-family ipv4
  interface all enable
  !
```

Verification

Verify that the Interfaces are enabled for multicast.

```
Router#show mfib interface location 0/3/CPU0
Interface : FINT0/3/CPU0 (Enabled)
SW Mcast pkts in : 0, SW Mcast pkts out : 0
TTL Threshold : 0
Ref Count : 2
Interface : TenGigE0/3/0/0/0 (Enabled)
SW Mcast pkts in : 0, SW Mcast pkts out : 0
TTL Threshold : 0
Ref Count : 3
Interface : TenGigE0/3/0/9/0 (Enabled)
SW Mcast pkts in : 0, SW Mcast pkts out : 0
TTL Threshold : 0
Ref Count : 13
Interface : Bundle-Ether1 (Enabled)
SW Mcast pkts in : 0, SW Mcast pkts out : 0
TTL Threshold : 0
Ref Count : 4
Interface : Bundle-Ether1.1 (Enabled)
SW Mcast pkts in : 0, SW Mcast pkts out : 0
TTL Threshold : 0
```

Supported Multicast Features

- Hardware Offloaded BFD for PIMv4 is supported.
- IPv4 and IPV6 static groups for both IGMPv2/v3 and MLDv1/v2 are supported.
- Protocol Independent Multicast in Source-Specific Multicast (PIM-SSM) mapping is supported.
- PIMv4 SSM over Bundle sub-interface is supported with the exception of PIMv6 SSM over Bundle sub-interface.
- Loadbalancing for multicast traffic for ECMP links and bundles is supported.
- Router needs to be reloaded to recover, if TCAM space is exceeded.
- Multicast MAC and multicast IP address should be matched for both Layer 2 and Layer 3 traffic, else traffic may be dropped by ASIC. L2 flooding is not supported.
- Multicast traffic fragmentation in hardware is not supported.
- Multicast traffic without Spanning-Tree protocol is supported at Layer 2 for multicast traffic without snooping enabled.
- IPv6 multicast MLD joins are subjected to hop by hop LPTS punt policer. Tweaking this policer to a higher value achieves convergence at higher scale.

Also, adjust the ICMP control traffic LPTS hardware policer to a higher value for optimal convergence at higher scale.

- Redundant sources for IPv6 PIM SSM is supported.

IGMP Snooping Features

Supported Features

- IGMP Snooping on bridge domain is supported
- Multicast on BVI is supported.
- EVPN IGMP State Sync using IGMP snooping profile is supported.

Restrictions for Multicast

- Multicast over core pseudowire is not supported.
- Multicast over access or core pseudowire is not supported.
- MFIB stats S,G not supported.
- The hw-module profile mfib statistics is not supported.
- IGMP snooping over VPLS is not supported.
- MLDP is not supported on the edge role.
- IPv6 PIM SM is not supported.
- AutoRP for IPv4 PIM SM is not supported.
- Static IPv4 mroutes are not supported.
- G.8032 or other L2 based redundancy and convergence protocols are not supported for multicast traffic.
- QoS over Multicast not supported.
- MVPN GRE is not supported.

Note You must enable **multicast-routing** in the default VRF to use multicast in some other VRF.

Restrictions for IGMP Snooping

- BVI enabled with Layer3 multicast and IGMP snooping is disabled.
- IGMP snooping reacts to only IGMP packets. PIM packets don't influence IGMP snooping decisions or status.
- Any *, G report IGMPv2/IGMPv3(exclude null) does not work. Sending *, G reports also breaks existing flows.

- PIMv4 Hello on BVI is does not gett punted due to punt code issue. Inject is fine.
- Different type of encapsulations such as dot1ad and qinq do not work for L2 subinterface attachment circuit. Encapsulation untagged is not supprted.
- BVI shutdown breaks punt path. Query Packets do not get punted.
- Egress traffic tagged with wrong encapsulation is not supported for traffic incoming to the BVI interface.
- Flooding on bridge domian when there is a BVI with no snooping profile attached is not supported on the access or P-Edge routers.
- IPv6 Multicast on BVI or pure bridge domain is not supported.

Protocol Independent Multicast

Protocol Independent Multicast (PIM) is a multicast routing protocol used to create multicast distribution trees, which are used to forward multicast data packets.

Proper operation of multicast depends on knowing the unicast paths towards a source or an RP. PIM relies on unicast routing protocols to derive this reverse-path forwarding (RPF) information. As the name PIM implies, it functions independently of the unicast protocols being used. PIM relies on the Routing Information Base (RIB) for RPF information. Protocol Independent Multicast (PIM) is designed to send and receive multicast routing updates.

NCS 5500 series router supports Protocol Independent Multicast in Source-Specific Multicast (PIM-SSM).

PIM on Bundle-Ethernet subinterface is supported.

PIM BFD Overview

The BFD Support for Multicast (PIM) feature, also known as PIM BFD, registers PIM as a client of BFD. PIM can then utilize BFD's fast adjacency failure detection. When PIM BFD is enabled, BFD enables faster failure detection without waiting for hello messages from PIM.

At PIMs request, as a BFD client, BFD establishes and maintains a session with an adjacent node for maintaining liveness and detecting forwarding path failure to the adjacent node. PIM hellos will continue to be exchanged between the neighbors even after BFD establishes and maintains a BFD session with the neighbor. The behavior of the PIM hello mechanism is not altered due to the introduction of this feature. Although PIM depends on the Interior Gateway Protocol (IGP) and BFD is supported in IGP, PIM BFD is independent of IGP's BFD.

Protocol Independent Multicast (PIM) uses a hello mechanism for discovering new PIM neighbors between adjacent nodes. The minimum failure detection time in PIM is 3 times the PIM Query-Interval. To enable faster failure detection, the rate at which a PIM hello message is transmitted on an interface is configurable. However, lower intervals increase the load on the protocol and can increase CPU and memory utilization and cause a system-wide negative impact on performance. Lower intervals can also cause PIM neighbors to expire frequently as the neighbor expiry can occur before the hello messages received from those neighbors are processed. When PIM BFD is enabled, BFD enables faster failure detection without waiting for hello messages from PIM.

Configure PIM BFD

This section describes how you can configure PIM BFD

```
Router# configure
Router(config)# router pim address-family ipv4
Router(config-pim-default-ipv4)# interface HundredGigE0/1/0/1
Router(config-pim-ipv4-if)#  bfd minimum-interval 10
Router(config-pim-ipv4-if)#  bfd fast-detect
Router(config-pim-ipv4-if)#  bfd multiplier 3
Router(config-pim-ipv4)#  exit
Router(config-pim-default-ipv4)# interface TenGigE0/0/0/4
Router(config-pim-ipv4-if)#  bfd minimum-interval 50
Router(config-pim-ipv4-if)#  bfd fast-detect
Router(config-pim-ipv4-if)#  bfd multiplier 3
Router(config-pim-ipv4-if)#  exit
Router(config-pim-default-ipv4)# interface TenGigE 0/0/0/4.101
Router(config-pim-ipv4-if)#  bfd minimum-interval 50
Router(config-pim-ipv4-if)#  bfd fast-detect
Router(config-pim-ipv4-if)#  bfd multiplier 3
Router(config-pim-ipv4-if)#  exit
Router(config-pim-default-ipv4)# interface  Bundle-Ether 101
Router(config-pim-ipv4-if)#  bfd minimum-interval 50
Router(config-pim-ipv4-if)#  bfd fast-detect
Router(config-pim-ipv4-if)#  bfd multiplier 3
Router(config-pim-ipv4-if)#  exit
Router(config-pim-default-ipv4)#  commit
```

Running Configuration

```
router pim
 address-family ipv4
  interface HundredGigE 0/1/0/1
   bfd minimum-interval 10
   bfd fast-detect
   bfd multiplier 3
  !
  interface TenGigE 0/0/0/4
   bfd minimum-interval 50
   bfd fast-detect
   bfd multiplier 3
  !
  interface TenGigE 0/0/0/4.101
   bfd minimum-interval 50
   bfd fast-detect
   bfd multiplier 3
  !
  interface Bundle-Ether 101
   bfd minimum-interval 50
   bfd fast-detect
   bfd multiplier 3
  !

  !
 !
!
```

Verification

The show outputs given in the following section display the details of the configuration of the PIM BFD, and the status of their configuration.

```
Router# show bfd session
Wed Nov 22 08:27:35.952 PST
Interface            Dest Addr    Local det time(int*mult)  State      Echo       Async
H/W     NPU
-------------------- --------------- ---------------- ---------------- -----      -----
-----    -----
Hu0/0/1/3            10.12.12.2      0s(0s*0) 90ms(30ms*3)     UP                 Yes
0/0/CPU0

Hu0/0/1/2            10.12.12.2      0s(0s*0) 90ms(30ms*3)     UP                 Yes
0/0/CPU0

Hu0/0/1/1            10.18.18.2      0s(0s*0) 90ms(30ms*3)     UP                 Yes
0/0/CPU0

Te0/0/0/4.101        10.112.112.2   0s(0s*0) 90ms(30ms*3)     UP                 Yes
0/0/CPU0

BE101                10.18.18.2      n/a      n/a              UP                 No    n/a


BE102                10.12.12.2      n/a      n/a              UP                 No    n/a



Router# show bfd client

Name             Node       Num sessions
---------------- ---------- --------------
L2VPN_ATOM       0/RP0/CPU0 0
MPLS-TR          0/RP0/CPU0 0
bgp-default      0/RP0/CPU0 0
bundlemgr_distrib 0/RP0/CPU0 14
isis-1           0/RP0/CPU0 0
object_tracking 0/RP0/CPU0 0
pim6             0/RP0/CPU0 0
pim              0/RP0/CPU0 0
service-layer    0/RP0/CPU0 0
```

Reverse Path Forwarding

Reverse-path forwarding (RPF) is an algorithm 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 an (S,G) entry present in the multicast routing table (a source-tree state), the router performs the RPF check against the IP address of the source for the multicast packet.
- If a PIM router has no explicit source-tree state, this is considered a shared-tree state. The router performs the RPF check on the address of the RP, 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.

Setting the Reverse Path Forwarding Statically

Configuration Example

The following example configures the static RPF rule for IP address 10.0.0.1:

```
Router#configure
Router(config)#multicast-routing
Router(config-mcast)#address-family ipv4
Router(config-mcast)#static-rpf 10.0.0.1 32 TenGigE 0/0/0/1 192.168.0.2
Router(config-mcast)#commit
```

Running Configuration

```
multicast-routing
   address-family ipv4
      static-rpf 10.10.10.2 32 TenGigE0/0/0/1 192.168.0.2
```

Verification

Verify that RPF is chosen according to the static RPF configuration for 10.10.10.2

```
Router#show pim rpf
Table: IPv4-Unicast-default
* 10.10.10.2/32 [0/0]
    via GigabitEthernet0/0/0/1 with rpf neighbor 192.168.0.2
```

RPF Vector Encoding Using IETF Standard

RPF vector is a PIM proxy that lets core routers without RPF information forward join and prune messages for external sources (for example, a MPLS-based BGP-free core, where the MPLS core router is without external routes learned from BGP). The RPF vector encoding is now compatible with the new IETF encoding. The new IETF standard encodes PIM messages using PIM Hello option 26.

Configuring RPF Vector (IETF Standard Encoding)

This example shows how to enable RPF encoding using IETF standard:

```
(config)# router pim
(config-pim-default-ipv4)# address-family ipv4
(config-pim-default-ipv4)# rpf-vector use-standard-encoding
!
(config)# multicast-routing
```

```
(config-mcast)# interface TenGigE0/1/0/7/0
(config-mcast)# interface TenGigE0/0/0/0/2
```

Verification

```
Router#show pim neighbor
Tue Apr 17 10:15:40.961 PDT

PIM neighbors in VRF default
Flag: B - Bidir capable, P - Proxy capable, DR - Designated Router,
      E - ECMP Redirect capable
      * indicates the neighbor created for this router

Neighbor Address             Interface              Uptime    Expires  DR pri   Flags

25.25.25.1                   TenGigE0/1/0/7/0       1w3d      00:01:36 1       B P
25.25.25.2*                  TenGigE0/1/0/7/0       1w3d      00:01:41 1 (DR) B P E
32.32.32.2*                  TenGigE0/0/0/0/2
       1w4d      00:01:40 1      B P E
32.32.32.3                   TenGigE0/0/0/0/2
       1w4d      00:01:42 1 (DR) B P
```

In the above output, you can see "P" tag on the multicast enabled interfaces.

PIM Bootstrap Router

The PIM bootstrap router (BSR) provides a fault-tolerant, automated RP discovery and distribution mechanism that simplifies the Auto-RP process. This feature is enabled by default allowing routers to dynamically learn the group-to-RP mappings.

PIM uses the BSR to discover and announce RP-set information for each group prefix to all the routers in a PIM domain. This is the same function accomplished by Auto-RP, but the BSR is part of the PIM specification. The BSR mechanism interoperates with Auto-RP on Cisco routers.

To avoid a single point of failure, you can configure several candidate BSRs in a PIM domain. A BSR is elected among the candidate BSRs automatically.

Candidates use bootstrap messages to discover which BSR has the highest priority. The candidate with the highest priority sends an announcement to all PIM routers in the PIM domain that it is the BSR.

Routers that are configured as candidate RPs unicast to the BSR the group range for which they are responsible. The BSR includes this information in its bootstrap messages and disseminates it to all PIM routers in the domain. Based on this information, all routers are able to map multicast groups to specific RPs. As long as a router is receiving the bootstrap message, it has a current RP map.

Configuring PIM Bootstrap Router

Configuration Example

Configures the router as a candidate BSR with a hash mask length of 30:

```
Router#config
Router(config)#router pim
Router(config-pim-default-ipv4)#bsr candidate-bsr 1.1.1.1 hash-mask-len 30 priority 1
Router(config-pim-default-ipv4-if)#commit
```

Configures the router to advertise itself as a candidate rendezvous point to the BSR in its PIM domain. Access list number 4 specifies the prefix associated with the candidate rendezvous point address 1.1.1.1 . This rendezvous point is responsible for the groups with the prefix 239.

```
Router#config
Router(config)#router pim
Router(config-pim-default-ipv4)#bsr candidate-rp 1.1.1.1 group-list 4 priority 192 interval
 60

Router(config-pim-default-ipv4)#exit
Router(config)#ipv4 access-list 4
Router(config-ipv4-acl)#permit ipv4 any 239.0.0.0 0.255.255.255
Router(config-ipv4-acl)#commit
```

Running Configuration

```
Router#show run router pim
router pim
 address-family ipv4
  bsr candidate-bsr 1.1.1.1 hash-mask-len 30 priority 1
  bsr candidate-rp 1.1.1.1 group-list 4 priority 192 interval 60
```

Verification

```
Router#show pim rp mapping
PIM Group-to-RP Mappings
Group(s) 239.0.0.0/8
  RP 1.1.1.1 (?), v2
    Info source: 1.1.1.1 (?), elected via bsr, priority 192, holdtime 150
      Uptime: 00:02:50, expires: 00:01:54

Router#show pim bsr candidate-rp
PIM BSR Candidate RP Info
Cand-RP          mode  scope priority uptime    group-list
1.1.1.1           BD    16      192       00:04:06    4

Router#show pim bsr election
PIM BSR Election State
Cand/Elect-State           Uptime   BS-Timer     BSR                                    C-BSR
Elected/Accept-Pref     00:03:49 00:00:25 1.1.1.1 [1, 30]          1.1.1.1 [1, 30]
```

PIM-Source Specific Multicast

When PIM is used in SSM mode, multicast routing is easier to manage. This is because RPs (rendezvous points) are not required and therefore, no shared trees (*,G) are built.

There is no specific IETF document defining PIM-SSM. However, RFC4607 defines the overall SSM behavior.

In the rest of this document, we use the term PIM-SSM to describe PIM behavior and configuration when SSM is used.

PIM in Source-Specific Multicast operation uses information found on source addresses for a multicast group provided by receivers and performs source filtering on traffic.

- By default, PIM-SSM operates in the 232.0.0.0/8 multicast group range for IPv4 and FF3x::/32 for IPv6. To configure these values, use the **ssm range** command.

Note PIM-SSM supports IPv6 from Cisco IOS XR Release 6.2.2

- If SSM is deployed in a network already configured for PIM-SM, only the last-hop routers must be upgraded with Cisco IOS XR Software that supports the SSM feature.
- No MSDP SA messages within the SSM range are accepted, generated, or forwarded.
- SSM can be disabled using the **ssm disable** command.
- The **ssm allow-override** command allows SSM ranges to be overridden by more specific ranges.

In many multicast deployments where the source is known, protocol-independent multicast-source-specific multicast (PIM-SSM) mapping is the obvious multicast routing protocol choice to use because of its simplicity. Typical multicast deployments that benefit from PIM-SSM consist of entertainment-type solutions like the ETTH space, or financial deployments that completely rely on static forwarding.

In SSM, delivery of data grams is based on (S,G) channels. Traffic for one (S,G) channel consists of datagrams with an IP unicast source address S and the multicast group address G as the IP destination address. Systems receive 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. Channel subscription signaling uses IGMP to include mode membership reports, which are supported only in Version 3 of IGMP (IGMPv3).

To run SSM with IGMPv3, SSM must be supported on the multicast router, the host where the application is running, and the application itself. Cisco IOS XR Software allows SSM configuration for an arbitrary subset of the IP multicast address range 224.0.0.0 through 239.255.255.255.

When an SSM range is defined, existing IP multicast receiver applications do not receive any traffic when they try to use addresses in the SSM range, unless the application is modified to use explicit (S,G) channel subscription.

Benefits of PIM-SSM over PIM-SM

PIM-SSM is derived from PIM-SM. However, whereas PIM-SM allows for the data transmission of all sources sending to a particular group in response to PIM join messages, the SSM feature forwards traffic to receivers only from those sources that the receivers have explicitly joined. Because PIM joins and prunes are sent directly towards the source sending traffic, an RP and shared trees are unnecessary and are disallowed. SSM is used to optimize bandwidth utilization and deny unwanted Internet broad cast traffic. The source is provided by interested receivers through IGMPv3 membership reports.

IGMPv2

To support IGMPv2, SSM mapping configuration must be added while configuring IGMP to match certain sources to group range.

Configuring Example

Configures the access-list (mc1):

```
Router#configure
Router(config)#ipv4 access-list mc1
Router(config-ipv4-acl)#permit ipv4 any 232.1.1.0 0.0.0.255
Router(config-ipv4-acl)#commit
```

Configures the multicast source (1.1.1.1) as part of a set of sources that map SSM groups described by the specified access-list (mc1):

```
Router#configure
Router(config)#router igmp
Router(config-igmp)#ssm map static 1.1.1.1 mc1
Router(config-igmp)#commit
```

Running Configuration

```
Router#show run router igmp
router igmp
ssm map static 1.1.1.1 mc1
```

Multipath Option

The multipath option is available under `router pim` configuration mode. After multipath option is enabled, SSM selects different path to reach same destination instead of choosing common path. The multipath option helps load balance the SSM traffic.

Configuring Multipath Option

```
Router#configure
Router(config)#router pim address-family ipv4
Router(config-pim-default-ipv4)#multipath hash source
Router(config-pim-default-ipv4)#commit
```

Running Configuration

```
Router#show running router pim
router pim
 address-family ipv4
  dr-priority 100
  multipath hash source    /*SSM traffic takes different path to reach same destination
based on source hash value.*/
```

Verification

The Bundle-Ether132 and TenGigE0/4/0/18/0.132 are two paths to reach the destination router Turnin-56. Since we have enabled multipath option, the source has two IP addresses 50.11.30.12 and 50.11.30.11. The Multicast traffic from two sources take two different paths Bundle-Ether132 and TenGigE0/4/0/18/0.132 to reach same destination.

This show run output shows that Bundle-Ether132 and TenGigE0/4/0/18/0.132 are connected to same destination router Turnin-56:

```
Router#show run int TenGigE0/1/0/6/3.132
interface TenGigE0/1/0/6/3.132
 description Connected to Turin-56 ten0/0/0/19.132
 ipv4 address 13.0.2.1 255.255.255.240
 ipv6 address 2606::13:0:2:1/120
 encapsulation dot1q 132
!

Router#show run int be132
interface Bundle-Ether132
 description Bundle between Fretta-56 and Turin-56
 ipv4 address 28.0.0.1 255.255.255.240
```

```
 ipv6 address 2606::28:0:0:1/120
 load-interval 30

Router#show mrib route 50.11.30.11 detail

IP Multicast Routing Information Base
Entry flags: L - Domain-Local Source, E - External Source to the Domain,
    C - Directly-Connected Check, S - Signal, IA - Inherit Accept,
    IF - Inherit From, D - Drop, ME - MDT Encap, EID - Encap ID,
    MD - MDT Decap, MT - MDT Threshold Crossed, MH - MDT interface handle
    CD - Conditional Decap, MPLS - MPLS Decap, EX - Extranet
    MoFE - MoFRR Enabled, MoFS - MoFRR State, MoFP - MoFRR Primary
    MoFB - MoFRR Backup, RPFID - RPF ID Set, X - VXLAN
Interface flags: F - Forward, A - Accept, IC - Internal Copy,
    NS - Negate Signal, DP - Don't Preserve, SP - Signal Present,
    II - Internal Interest, ID - Internal Disinterest, LI - Local Interest,
    LD - Local Disinterest, DI - Decapsulation Interface
    EI - Encapsulation Interface, MI - MDT Interface, LVIF - MPLS Encap,
    EX - Extranet, A2 - Secondary Accept, MT - MDT Threshold Crossed,
    MA - Data MDT Assigned, LMI - mLDP MDT Interface, TMI - P2MP-TE MDT Interface
    IRMI - IR MDT Interface

(50.11.30.11,225.255.11.1) Ver: 0x523cc294 RPF nbr: 50.11.30.11 Flags: L RPF, FGID: 11453,
 -1, -1
  Up: 4d15h
  Incoming Interface List
    HundredGigE0/4/0/10.1130 Flags: A, Up: 4d15h
  Outgoing Interface List
    FortyGigE0/1/0/5 Flags: F NS, Up: 4d15h
    TenGigE0/4/0/6/0 Flags: F NS, Up: 4d15h
    TenGigE0/1/0/6/3.132 Flags: F NS, Up: 4d15h
    TenGigE0/4/0/18/0.122 Flags: F NS, Up: 4d15h

Router#show mrib route 50.11.30.12 detail

IP Multicast Routing Information Base
Entry flags: L - Domain-Local Source, E - External Source to the Domain,
    C - Directly-Connected Check, S - Signal, IA - Inherit Accept,
    IF - Inherit From, D - Drop, ME - MDT Encap, EID - Encap ID,
    MD - MDT Decap, MT - MDT Threshold Crossed, MH - MDT interface handle
    CD - Conditional Decap, MPLS - MPLS Decap, EX - Extranet
    MoFE - MoFRR Enabled, MoFS - MoFRR State, MoFP - MoFRR Primary
    MoFB - MoFRR Backup, RPFID - RPF ID Set, X - VXLAN
Interface flags: F - Forward, A - Accept, IC - Internal Copy,
    NS - Negate Signal, DP - Don't Preserve, SP - Signal Present,
    II - Internal Interest, ID - Internal Disinterest, LI - Local Interest,
    LD - Local Disinterest, DI - Decapsulation Interface
    EI - Encapsulation Interface, MI - MDT Interface, LVIF - MPLS Encap,
    EX - Extranet, A2 - Secondary Accept, MT - MDT Threshold Crossed,
    MA - Data MDT Assigned, LMI - mLDP MDT Interface, TMI - P2MP-TE MDT Interface
    IRMI - IR MDT Interface

(50.11.30.12,226.255.12.1) Ver: 0x5fe02e5b RPF nbr: 50.11.30.12 Flags: L RPF, FGID: 12686,
 -1, -1
  Up: 4d15h
  Incoming Interface List
    HundredGigE0/4/0/10.1130 Flags: A, Up: 4d15h
  Outgoing Interface List
    Bundle-Ether121 Flags: F NS, Up: 4d15h
    Bundle-Ether132 Flags: F NS, Up: 4d15h
    FortyGigE0/1/0/5 Flags: F NS, Up: 4d15h
    TenGigE0/4/0/6/0.117 Flags: F NS, Up: 4d15h
```

Configuring PIM-SSM

Configuration Example

Configures SSM service for the IPv4 address range defined by access list 4.

```
Router#config
Router(config)#ipv4 access-list 4
Router(config-ipv4-acl)#permit ipv4 any 224.2.151.0 0.0.0.255
Router(config-ipv4-acl)#exit
Router(config)#multicast-routing
Router(config-mcast)#address-family ipv4
Router(config-mcast-default-ipv4)#ssm range 4
Router(config-mcast-default-ipv4)#commit
Router(config-mcast-default-ipv4)#end
```

Configures SSM service for the IPv6 address range defined by access list 6.

```
Router#config
Router(config)#ipv6 access-list 6
Router(config-ipv6-acl)#permit ipv6 any ff30:0:0:2::/32
Router(config-ipv6-acl)#exit
Router(config)#multicast-routing
Router(config-mcast)#address-family ipv6
Router(config-mcast-default-ipv6)#ssm range 6
Router(config-mcast-default-ipv6)#commit
Router(config-mcast-default-ipv6)#end
```

Running Configuration

```
Router#show running multicast-routing
multicast-routing
 address-family ipv4
  ssm range 4
  interface all enable
 !
```

```
Router#show running multicast-routing
multicast-routing
 address-family ipv6
  ssm range 6
  interface all enable
 !
```

Verification

Verify if the SSM range is configured according to the set parameters:

```
Router#show access-lists 4
ipv4 access-list 4
 10 permit ipv4 any 224.2.151.0 0.0.0.255

*/Verify if the SSM is configured for 224.2.151.0/24/*:

Router#show pim group-map
IP PIM Group Mapping Table
(* indicates group mappings being used)
Group Range         Proto Client   Groups RP address      Info
224.0.1.39/32*      DM    perm     1      0.0.0.0
224.0.1.40/32*      DM    perm     1      0.0.0.0
224.0.0.0/24*       NO    perm     0      0.0.0.0
224.2.151.0/24*     SSM   config   0      0.0.0.0
```

Configuring PIM Parameters

To configure PIM-specific parameters, the router pim configuration mode is used. The default configuration prompt is for IPv4 and will be seen as config-pim-default-ipv4. To ensure the election of a router as PIM DR on a LAN segment, use the **dr-priority** command. The router with the highest DR priority will win the election. By default, at a preconfigured threshold, the last hop router can join the shortest path tree to receive multicast traffic. To change this behavior, use the command **spt-threshold infinity** under the router pim configuration mode. This will result in the last hop router permanently joining the shared tree. The frequency at which a router sends PIM hello messages to its neighbors can be configured by the hello-interval command. By default, PIM hello messages are sent once every 30 seconds. If the hello-interval is configured under router pim configuration mode, all the interfaces with PIM enabled will inherit this value. To change the hello interval on the interface, use the **hello-interval** command under interface configuration mode, as follows:

Configuration Example

```
Router#configure
Router(config)#router pim
Router(config-pim-default)#address-family ipv4
Router(config-pim-default-ipv4)#dr-priority 2
Router(config-pim-default-ipv4)#spt-threshold infinity
Router(config-pim-default-ipv4)#interface TenGigE0/0/0/1
Router(config-pim-ipv4-if)#dr-priority 4
Router(config-pim-ipv4-if)#hello-interval 45
Router(config-pim-ipv4-if)#commit
```

Running Configuration

```
Router#show run router pim
router pim
 address-family ipv4
  dr-priority 2
  spt-threshold infinity
  interface TenGigE0/0/0/1
   dr-priority 4
   hello-interval 45
```

Verification

Verify if the parameters are set according to the configured values:

```
Router#show pim interface te0/0/0/1
PIM interfaces in VRF default
Address               Interface                PIM  Nbr   Hello  DR    DR Count Intvl
  Prior
100.1.1.1             TenGigE0/0/0/1           on   1     45     4     this system
```

Multicast Source Discovery Protocol

Multicast Source Discovery Protocol (MSDP) is a mechanism to connect multiple PIM sparse-mode domains. MSDP allows multicast sources for a group to be known to all rendezvous points (RPs) in different domains. Each PIM-SM domain uses its own RPs and need not depend on RPs in other domains.

An RP in a PIM-SM domain has MSDP peering relationships with MSDP-enabled routers in other domains. Each peering relationship occurs over a TCP connection, which is maintained by the underlying routing system.

MSDP speakers exchange messages called Source Active (SA) messages. When an RP learns about a local active source, typically through a PIM register message, the MSDP process encapsulates the register in an SA message and forwards the information to its peers. The message contains the source and group information for the multicast flow, as well as any encapsulated data. If a neighboring RP has local joiners for the multicast group, the RP installs the S, G route, forwards the encapsulated data contained in the SA message, and sends PIM joins back towards the source. This process describes how a multicast path can be built between domains.

Note Although you should configure BGP or Multiprotocol BGP for optimal MSDP interdomain operation, this is not considered necessary in the Cisco IOS XR Software implementation. For information about how BGP or Multiprotocol BGP may be used with MSDP, see the MSDP RPF rules listed in the Multicast Source Discovery Protocol (MSDP), Internet Engineering Task Force (IETF) Internet draft.

MSDP Configuration Submode

When you issue the **router msdp** command, the CLI prompt changes to "config-msdp," indicating that you have entered router MSDP configuration submode.

Interconnecting PIM-SM Domains with MSDP

To set up an MSDP peering relationship with MSDP-enabled routers in another domain, you configure an MSDP peer to the local router.

If you do not want to have or cannot have a BGP peer in your domain, you could define a default MSDP peer from which to accept all Source-Active (SA) messages.

Finally, you can change the Originator ID when you configure a logical RP on multiple routers in an MSDP mesh group.

Before you begin

You must configure MSDP default peering, if the addresses of all MSDP peers are not known in BGP or multiprotocol BGP.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ipv4 address** *address mask*
4. **exit**
5. **router msdp**
6. **default-peer** *ip-address* [**prefix-list** *list*]
7. **originator-id** *type interface-path-id*
8. **peer** *peer-address*
9. **connect-source** *type interface-path-id*
10. **mesh-group** *name*
11. **remote-as** *as-number*
12. **commit**
13. **show msdp** [**ipv4**] **globals**

14. **show msdp** [**ipv4**] **peer** [*peer-address*]
15. **show msdp** [**ipv4**] **rpf** *rpf-address*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**interface** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config)# interface loopback 0`	(Optional) Enters interface configuration mode to define the IPv4 address for the interface. **Note** This step is required if you specify an interface type and number whose primary address becomes the source IP address for the TCP connection.
Step 3	**ipv4 address** *address mask* **Example:** `RP/0/RP0/CPU0:router(config-if)# ipv4 address 10.0.1.3 255.255.255.0`	(Optional) Defines the IPv4 address for the interface. **Note** This step is required only if you specify an interface type and number whose primary address becomes the source IP address for the TCP connection. See optional for information about configuring the **connect-source** command.
Step 4	**exit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end`	Exits interface configuration mode.
Step 5	**router msdp** **Example:** `RP/0/RP0/CPU0:router(config)# router msdp`	Enters MSDP protocol configuration mode.
Step 6	**default-peer** *ip-address* [**prefix-list** *list*] **Example:** `RP/0/RP0/CPU0:router(config-msdp)# default-peer 172.23.16.0`	(Optional) Defines a default peer from which to accept all MSDP SA messages.
Step 7	**originator-id** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config-msdp)# originator-id /1/1/0`	(Optional) Allows an MSDP speaker that originates a (Source-Active) SA message to use the IP address of the interface as the RP address in the SA message.

	Command or Action	Purpose
Step 8	**peer** *peer-address* **Example:** `RP/0/RP0/CPU0:router(config-msdp)# peer 172.31.1.2`	Enters MSDP peer configuration mode and configures an MSDP peer. • Configure the router as a BGP neighbor. • If you are also BGP peering with this MSDP peer, use the same IP address for MSDP and BGP. You are not required to run BGP or multiprotocol BGP with the MSDP peer, as long as there is a BGP or multiprotocol BGP path between the MSDP peers.
Step 9	**connect-source** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config-msdp-peer)# connect-source loopback 0`	(Optional) Configures a source address used for an MSDP connection.
Step 10	**mesh-group** *name* **Example:** `RP/0/RP0/CPU0:router(config-msdp-peer)# mesh-group internal`	(Optional) Configures an MSDP peer to be a member of a mesh group.
Step 11	**remote-as** *as-number* **Example:** `RP/0/RP0/CPU0:router(config-msdp-peer)# remote-as 250`	(Optional) Configures the remote autonomous system number of this peer.
Step 12	**commit**	
Step 13	**show msdp** [**ipv4**] **globals** **Example:** `RP/0/RP0/CPU0:router# show msdp globals`	Displays the MSDP global variables.
Step 14	**show msdp** [**ipv4**] **peer** [*peer-address*] **Example:** `RP/0/RP0/CPU0:router# show msdp peer 172.31.1.2`	Displays information about the MSDP peer.
Step 15	**show msdp** [**ipv4**] **rpf** *rpf-address* **Example:** `RP/0/RP0/CPU0:router# show msdp rpf 172.16.10.13`	Displays the RPF lookup.

Controlling Source Information on MSDP Peer Routers

Your MSDP peer router can be customized to control source information that is originated, forwarded, received, cached, and encapsulated.

When originating Source-Active (SA) messages, you can control to whom you will originate source information, based on the source that is requesting information.

When forwarding SA messages you can do the following:

- Filter all source/group pairs
- Specify an extended access list to pass only certain source/group pairs
- Filter based on match criteria in a route map

When receiving SA messages you can do the following:

- Filter all incoming SA messages from an MSDP peer
- Specify an extended access list to pass certain source/group pairs
- Filter based on match criteria in a route map

In addition, you can use time to live (TTL) to control what data is encapsulated in the first SA message for every source. For example, you could limit internal traffic to a TTL of eight hops. If you want other groups to go to external locations, you send those packets with a TTL greater than eight hops.

By default, MSDP automatically sends SA messages to peers when a new member joins a group and wants to receive multicast traffic. You are no longer required to configure an SA request to a specified MSDP peer.

SUMMARY STEPS

1. **configure**
2. **router msdp**
3. **sa-filter** {**in** | **out**} {*ip-address* | *peer-name*} [**list** *access-list-name*] [**rp-list** *access-list-name*]
4. **cache-sa-state** [**list** *access-list-name*] [**rp-list** *access-list-name*]
5. **ttl-threshold** *ttl-value*
6. **exit**
7. **ipv4 access-list** *name* [*sequence-number*] **permit** *source* [*source-wildcard*]
8. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**router msdp** **Example:** `RP/0/RP0/CPU0:router(config)# router msdp`	Enters MSDP protocol configuration mode.

	Command or Action	Purpose
Step 3	**sa-filter** {**in** | **out**} {*ip-address* | *peer-name*} [**list** *access-list-name*] [**rp-list** *access-list-name*] **Example:** `RP/0/RP0/CPU0:router(config-msdp)# sa-filter out router.cisco.com list 100`	Configures an incoming or outgoing filter list for messages received from the specified MSDP peer. • If you specify both the **list** and **rp-list** keywords, all conditions must be true to pass any source, group (S, G) pairs in outgoing Source-Active (SA) messages. • You must configure the **ipv4 access-list** command in Step 7, on page 20. • If all match criteria are true, a **permit** from the route map passes routes through the filter. A **deny** filters routes. • This example allows only (S, G) pairs that pass access list 100 to be forwarded in an SA message to the peer named router.cisco.com.
Step 4	**cache-sa-state** [**list** *access-list-name*] [**rp-list** *access-list-name*] **Example:** `RP/0/RP0/CPU0:router(config-msdp)# cache-sa-state list 100`	Creates and caches source/group pairs from received Source-Active (SA) messages and controls pairs through access lists.
Step 5	**ttl-threshold** *ttl-value* **Example:** `RP/0/RP0/CPU0:router(config-msdp)# ttl-threshold 8`	(Optional) Limits which multicast data is sent in SA messages to an MSDP peer. • Only multicast packets with an IP header TTL greater than or equal to the *ttl-value* argument are sent to the MSDP peer specified by the IP address or name. • Use this command if you want to use TTL to examine your multicast data traffic. For example, you could limit internal traffic to a TTL of 8. If you want other groups to go to external locations, send those packets with a TTL greater than 8. • This example configures a TTL threshold of eight hops.
Step 6	**exit** **Example:** `RP/0/RP0/CPU0:router(config-msdp)# exit`	Exits the current configuration mode.
Step 7	**ipv4 access-list** *name* [*sequence-number*] **permit** *source* [*source-wildcard*] **Example:**	Defines an IPv4 access list to be used by SA filtering. • In this example, the access list 100 permits multicast group 239.1.1.1.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router(config)# ipv4 access-list 100 20 permit 239.1.1.1 0.0.0.0`	• The **ipv4 access-list** command is required if the keyword **list** is configured for SA filtering in Step 3, on page 20.
Step 8	**commit**	

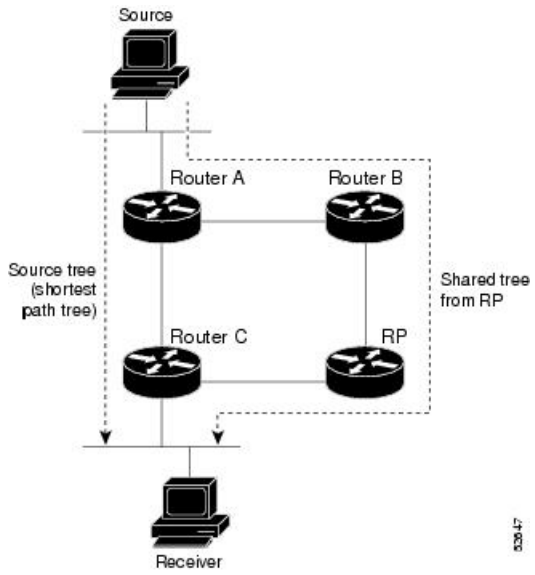
PIM-Sparse Mode

Typically, PIM in sparse mode (PIM-SM) operation is used in a multicast network when relatively few routers are involved in each multicast. Routers do not forward multicast packets for a group, unless there is an explicit request for traffic. Requests are accomplished using PIM join messages, which are sent hop by hop toward the root node of the tree. The root node of a tree in PIM-SM is the rendezvous point (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 sources that send multicast packets are registered with the RP by the first-hop router of the source.

As a PIM join travels up the tree, routers along the path set up the multicast forwarding state so that the requested multicast traffic is forwarded back down the tree. When multicast traffic is no longer needed, a router sends a PIM prune message 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. Additionally, if prunes are not explicitly sent, the PIM state will timeout and be removed in the absence of any further join messages.

This image shows IGMP and PIM-SM operating in a multicast environment.

Figure 1: Shared Tree and Source Tree (Shortest Path Tree)



In PIM-SM, the rendezvous point (RP) is used to bridge sources sending data to a particular group with receivers sending joins for that group. In the initial set up of state, interested receivers receive data from senders to the group across a single data distribution tree rooted at the RP. This type of distribution tree is called a shared tree or rendezvous point tree (RPT) as illustrated in Figure 4: Shared Tree and Source Tree

(Shortest Path Tree), above. Data from senders is delivered to the RP for distribution to group members joined to the shared tree.

Unless the command is configured, this initial state gives way as soon as traffic is received on the leaf routers (designated router closest to the host receivers). When the leaf router receives traffic from the RP on the RPT, the router initiates a switch to a data distribution tree rooted at the source sending traffic. This type of distribution tree is called a shortest path tree or source tree. By default, the Cisco IOS XR Software switches to a source tree when it receives the first data packet from a source.

The following process describes the move from shared tree to source tree in more detail:

1. Receiver joins a group; leaf Router C sends a join message toward RP.
2. RP puts link to Router C in its outgoing interface list.
3. Source sends data; Router A encapsulates data in Register and sends it to RP.
4. RP forwards data down the shared tree to Router C and sends a join message toward Source. At this point, data may arrive twice at the RP, once encapsulated and once natively.
5. When data arrives natively (unencapsulated) at RP, 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 Source.
7. When Router C receives data on (S,G), it sends a prune message for Source up the shared tree.
8. RP deletes the link to Router C from outgoing interface of (S,G). RP triggers a prune message toward Source.
9. 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 exchanged using direct unicast communication between the designated router that is directly connected to a source and the RP for the group.

Note The **spt-threshold infinity** command lets you configure the router so that it never switches to the shortest path tree (SPT).

Restrictions and Usage Guidelines

This section describes the restrictions and guidelines related to the PIM protocol.

The following restrictions and guidelines apply for the Protocol Independent Multicast-Sparse Mode (PIM-SM) protocol:

- Only IPv4 is supported; IPv6 is not supported.
- For native multicast, only mVPN GRE (applicable for both the NC55 and NC57 line cards) and profile 14 are supported.
- Only Static Rendezvous Point (RP) is supported; Auto RP is not supported.
- PIM Bootstrap Router (BSR) is not supported for Profile 14.

Designated Routers

Cisco routers use PIM-SM to forward multicast traffic and follow an election process to select a designated router (DR) when there is more than one router on a LAN segment.

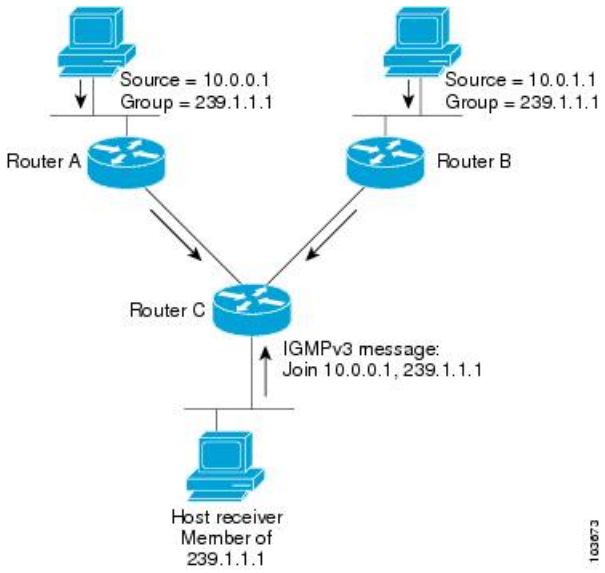
The designated router is responsible for sending PIM register and PIM join and prune messages toward the RP to inform it about host group membership.

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 IP address becomes the DR for the LAN unless you choose to force the DR election by use of the **dr-priority** command. The DR priority option 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 is elected as the DR. If all routers on the LAN segment have the same priority, the highest IP address is again used as the tiebreaker.

Note DR election process is required only on multi access LANs. The last-hop router directly connected to the host is the DR.

The figure "Designated Router Election on a Multiaccess Segment", below illustrates what happens on a multi access segment. Router A (10.0.0.253) and Router B (10.0.0.251) are connected to a common multi access Ethernet segment with Host A (10.0.0.1) as an active receiver for Group A. As the Explicit Join model is used, only Router A, operating as the DR, sends joins to the RP to construct the shared tree for Group A. If Router B were also permitted to send (*,G) joins to the RP, parallel paths would be created and Host A would receive duplicate multicast traffic. When Host A begins to source multicast traffic to the group, the DR's responsibility is to send register messages to the RP. Again, if both routers were assigned the responsibility, the RP would receive duplicate multicast packets.

Figure 2: Designated Router Election on a Multiaccess Segment



If the DR fails, the PIM-SM provides a way to detect the failure of Router A and to elect a failover DR. If the DR (Router A) were to become inoperable, Router B would detect this situation when its neighbor adjacency

with Router A timed out. Because Router B has been hearing IGMP membership reports from Host A, it already has IGMP state for Group A on this interface and immediately sends a join to the RP when it becomes the new DR. This step reestablishes traffic flow down a new branch of the shared tree using 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, using a new branch through Router B.

Note Two PIM routers are neighbors if there is a direct connection between them. To display your PIM neighbors, use the show pim neighbor command in EXEC mode.

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

Configuration Example

Configures the router to use DR priority 4 for TenGigE interface 0/0/0/1, but other interfaces will inherit DR priority 2:

```
Router#configure
Router(config)#router pim
Router(config-pim-default)#address-family ipv4
Router(config-pim-default-ipv4)#dr-priority 2
Router(config-pim-default-ipv4)#interface TenGigE0/0/0/1
Router(config-pim-ipv4-if)#dr-priority 4
Router(config-ipv4-acl)#commit
```

Running Configuration

```
Router#show run router pim
router pim
 address-family ipv4
  dr-priority 2
  spt-threshold infinity
  interface TenGigE0/0/0/1
   dr-priority 4
   hello-interval 45
```

Verification

Verify if the parameters are set according to the configured values:

```
Router#show pim interface
PIM interfaces in VRF default
Address           Interface           PIM  Nbr   Hello  DR    DR Count Intvl  Prior
100.1.1.1         TenGigE0/0/0/1       on   1     45     4      this system
26.1.1.1          TenGigE0/0/0/26      on   1     30     2      this system
```

Designated Router Election Using StickyDR

Table 1: Feature History Table

Feature Name	Release Information	Feature Description
Designated Router Election Using StickyDR	Release 7.4.1	With this feature, the router sends a PIM *hello* message with a special PIM DR priority value on a multi-access LAN. The router with this special DR priority value is always elected as the designated router. The traffic now flows in the same path even when a new router is added. This feature introduces the **sticky-dr** command.

When you enable PIM on an interface or reload a router, router periodically sends the PIM Hello messages on each interface. PIM Hello messages allow a router to learn neighboring PIM routers on each interface and elects a Designated Router (DR) based on the DR Priority. The DR election avoids duplicating multicast traffic for connected hosts.

Each time the DR is reelected, the multicast control tree sets up a new path and the multicast traffic flows in different direction.

With Sticky DR feature, the designated router remains the same and doesn't allow any other router to become the designated router. The multicast control tree does not set up a new path and the multicast traffic flows in same direction, thus avoids traffic loss. DR election isn't based on DR priority.

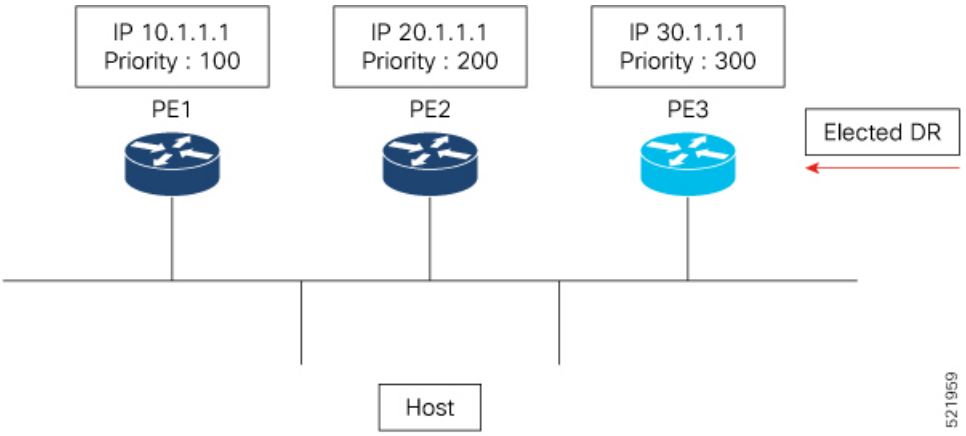
After you enable the sticky DR feature, the elected DR no longer advertises configured DR. Instead the router sends PIM Hello message with special PIM DR priority value which is reserved for Sticky PIM DR.

Restrictions

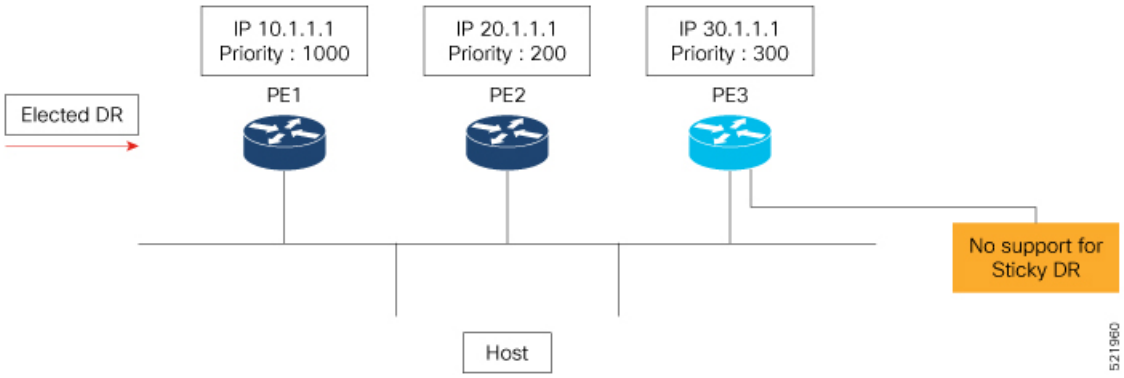
- The Sticky DR priority value is 4294967294. You must not configure DR priority with the value 4294967294 or any number greater than this value.

Topology

In this topology, PE1, PE2, PE3 are three PIM routers connected on a LAN. PE3 has the maximum priority and hence PE3 is elected as DR.

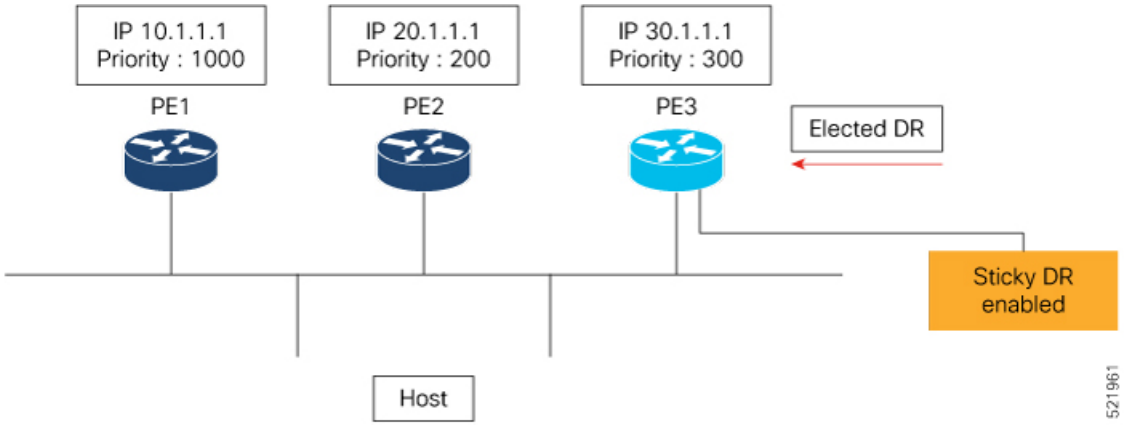


Now, when you configure PE1 with DR priority 1000, DR election process is re-initiated and PE1 becomes the new DR.



Every time a new DR is elected, the control tree computes a new path for traffic flow.

Now if you enable sticky DR on PE3, the PE3 remains the designated router irrespective of the DR priority of the PE devices.



In this example, the sticky DR is configured on PE3 and PE3 always remains as the DR.

Configuration

Let's configure sticky DR on PE3. To configure sticky DR on an interface, perform the following task:

```
Router# configure
Router(config)# router pim
Router(config-pim-default)# address-family ipv4
Router(config-pim-default-ipv4)# interface bundle-ether 72.1
Router(config-pim-ipv4-if)# sticky-dr
Router(config-ipv4-acl)# commit
```

Verification

The following output specifies that the Sticky DR is enabled on the interface and active:

```
Router# show pim interface bundle-ether 72.1 detail

PIM interfaces in VRF default
IP PIM Multicast Interface State
Flag: B - Bidir enabled, NB - Bidir disabled
      P - PIM Proxy enabled, NP - PIM Proxy disabled
      V - Virtual Interface, S - Sticky DR enabled
BFD State - State/Interval/Multiplier

Interface                  PIM  Nbr   Hello  DR
                                Count Intvl  Prior

Bundle-Ether72.1            on   2     30     100000
    Primary Address : 200.1.72.1
              Flags : B NP S V
                BFD : On/150 ms/3
                 DR : this system
  Propagation delay : 500
  Override Interval : 2500
        Hello Timer : 00:00:24
    Neighbor Filter : -
          Sticky DR : Configured, Active since Mon Jul 26 16:53:01 2021

-----------------------
Sticky DR Event History
-----------------------
Event                     State        Time
-----                     -----        ----
Dynamic Batch             Active       (null)
```

The following output specifies that the Sticky DR is enabled on the interface and is inactive:

```
Router#  show pim interface bundle-ether 72.1 detail

PIM interfaces in VRF default
IP PIM Multicast Interface State
Flag: B - Bidir enabled, NB - Bidir disabled
      P - PIM Proxy enabled, NP - PIM Proxy disabled
      V - Virtual Interface, S - Sticky DR enabled
BFD State - State/Interval/Multiplier

Interface                  PIM  Nbr   Hello  DR
                                Count Intvl  Prior

Bundle-Ether72.1            on   2     30     1
    Primary Address : 200.1.72.1
              Flags : B NP S V
                BFD : On/150 ms/3
```

```
               DR : 200.1.72.2
  Propagation delay : 500
  Override Interval : 2500
        Hello Timer : 00:00:18
    Neighbor Filter : -
          Sticky DR : Configured, Inactive
```

```
Router# show pim neighor detail

PIM neighbors in VRF default
Flag: B - Bidir capable, P - Proxy capable, DR - Designated Router,
      E - ECMP Redirect capable, S - Sticky DR Neighbor
      * indicates the neighbor created for this router

Neighbor Address             Interface              Uptime    Expires  DR pri      Flags

201.7.7.7*                   tunnel-mte1019         2d17h     00:01:36 1           (DR) B
E
  Expiry Timer: 00:01:05
201.7.7.7*                   tunnel-mte1001         2d17h     00:01:36 1           (DR) B
E
  Expiry Timer: 00:01:12
200.1.71.1*                  Bundle-Ether71.1       2d17h     00:01:31 99          (DR) B
  Expiry Timer: 00:00:02
200.1.71.2                   Bundle-Ether71.1       2d17h     00:01:19 1           B
BFD State: enabled
201.7.7.7*                   Loopback0              2d17h     00:01:41 1           (DR) B
E
  Expiry Timer: 00:01:12
201.202.7.7*                 Loopback1              2d17h     00:01:40 1           (DR) B
E
  Expiry Timer: 00:01:11
200.1.72.1*                  Bundle-Ether72.1       2d17h     00:01:15 -           (DR) B
S
  Expiry Timer: 00:01:21
```

Disable Sticky DR

To disable the sticky DR feature, perform the following task:

```
Router# configure
Router(config)# router pim
Router(config-pim-default)# address-family ipv4
Router(config-pim-default-ipv4)# interface bundle-ether 72.1
Router(config-pim-ipv4-if)# no sticky-dr
Router(config-ipv4-acl)# commit
```

To clear the DR stickiness and force the DR reelection, use the following command:

```
Router# clear pim interface bundle-ether 72.1 sticky-dr
```

Multicast VPN

Multicast VPN (MVPN) provides the ability to dynamically provide multicast support over MPLS networks. MVPN introduces an additional set of protocols and procedures that help enable a provider to support multicast traffic in a VPN.

Note PIM-Bidir is not supported on MVPN.

There are two ways MCAST VPN traffic can be transported over the core network:

- Rosen GRE (native): MVPN uses GRE with unique multicast distribution tree (MDT) forwarding to enable scalability of native IP Multicast in the core network. MVPN introduces multicast routing information to the VPN routing and forwarding table (VRF), creating a Multicast VRF. In Rosen GRE, the MCAST customer packets (c-packets) are encapsulated into the provider MCAST packets (p-packets), so that the PIM protocol is enabled in the provider core, and mrib/mfib is used for forwarding p-packets in the core.
- MLDP ones (Rosen, partition): MVPN allows a service provider to configure and support multicast traffic in an MPLS VPN environment. This type supports routing and forwarding of multicast packets for each individual VPN routing and forwarding (VRF) instance, and it also provides a mechanism to transport VPN multicast packets across the service provider backbone. In the MLDP case, the regular label switch path forwarding is used, so core does not need to run PIM protocol. In this scenario, the c-packets are encapsulated in the MPLS labels and forwarding is based on the MPLS Label Switched Paths (LSPs) ,similar to the unicast case.

In both the above types, the MVPN service allows you to build a Protocol Independent Multicast (PIM) domain that has sources and receivers located in different sites.

To provide Layer 3 multicast services to customers with multiple distributed sites, service providers look for a secure and scalable mechanism to transmit customer multicast traffic across the provider network. Multicast VPN (MVPN) provides such services over a shared service provider backbone, using native multicast technology similar to BGP/MPLS VPN.

MVPN emulates MPLS VPN technology in its adoption of the multicast domain (MD) concept, in which provider edge (PE) routers establish virtual PIM neighbor connections with other PE routers that are connected to the same customer VPN. These PE routers thereby form a secure, virtual multicast domain over the provider network. Multicast traffic is then transmitted across the core network from one site to another, as if the traffic were going through a dedicated provider network.

Multi-instance BGP is supported on multicast and MVPN. Multicast-related SAFIs can be configured on multiple BGP instances.

Multicast VPN Routing and Forwarding

Dedicated multicast routing and forwarding tables are created for each VPN to separate traffic in one VPN from traffic in another.

The VPN-specific multicast routing and forwarding database is referred to as **MVRF**. On a PE router, an MVRF is created when multicast is enabled for a VRF. Protocol Independent Multicast (PIM), and Internet Group Management Protocol (IGMP) protocols run in the context of MVRF, and all routes created by an MVRF protocol instance are associated with the corresponding MVRF. In addition to VRFs, which hold VPN-specific protocol states, a PE router always has a global VRF instance, containing all routing and forwarding information for the provider network.

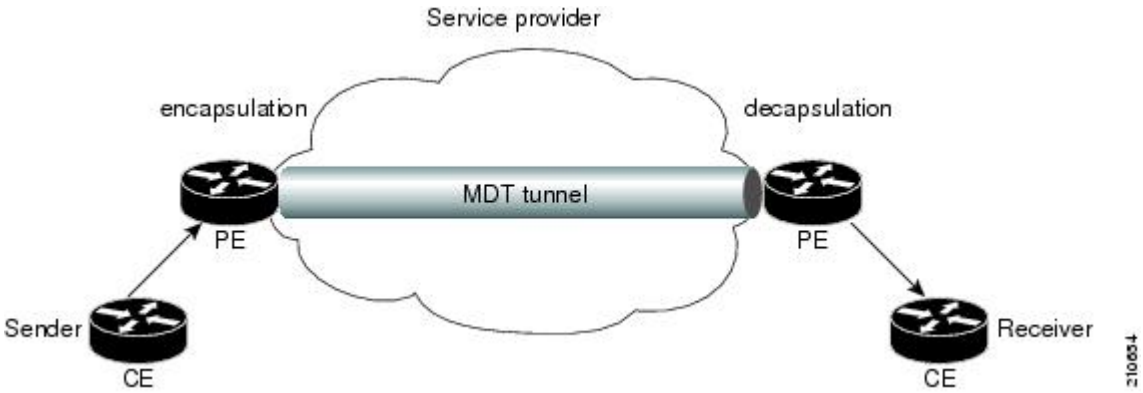
Multicast Distribution Tree Tunnels

The multicast distribution tree (MDT) can span multiple customer sites through provider networks, allowing traffic to flow from one source to multiple receivers. For MLDP, the MDT tunnel are called Labeled MDT (LMDT).

Secure data transmission of multicast packets sent from the customer edge (CE) router at the ingress PE router is achieved by encapsulating the packets in a provider header and transmitting the packets across the core. At the egress PE router, the encapsulated packets are decapsulated and then sent to the CE receiving routers.

Multicast distribution tree (MDT) tunnels are point-to-multipoint. A MDT tunnel interface is an interface that MVRF uses to access the multicast domain. It can be deemed as a passage that connects an MVRF and the global MVRF. Packets sent to an MDT tunnel interface are received by multiple receiving routers. Packets sent to an MDT tunnel interface are encapsulated, and packets received from a MDT tunnel interface are decapsulated.

Figure 3: Virtual PIM Peer Connection over an MDT Tunnel Interface



Encapsulating multicast packets in a provider header allows PE routers to be kept unaware of the packets' origin—all VPN packets passing through the provider network are viewed as native multicast packets and are routed based on the routing information in the core network. To support MVPN, PE routers only need to support native multicast routing.

MVPN also supports optimized VPN traffic forwarding for high-bandwidth applications that have sparsely distributed receivers. A dedicated multicast group can be used to encapsulate packets from a specific source, and an optimized MDT can be created to send traffic only to PE routers connected to interested receivers. This is referred to **data MDT**.

Naming Data MDTs

Table 2: Feature History Table

Feature Name	Release Information	Feature Description
Naming Data MDTs	Release 7.5.2	You can decide on the multicast flow that must be mapped to a specific MDT by naming it and assigning the flows to the named data MDT.

This feature enables you to deterministically map flows to data MDT at ingress PE by assigning the flows to a named data MDT.

You can define and create policies to map flows, which must flow to the required PEs, to specific data MDTs instead of dynamically multiplexing the flow to existing data MDT. This mapping makes the distribution of flows to data MDT more deterministic and solves the inherent inefficiencies.

Transitioning a flow to a data MDT occurs in one of the following conditions:

- The rate of the flow reaches the configured threshold limit.

  You can set the threshold limit of data MDTs by using the following command:

  ```
  mdt data <max nr of data groups> (threshold)
  ```

- Immediate switch is configured in which case the flow when detected is transitioned to a data MDT.

  For more details of immediate switch option configuration, see mdt data ingress replication command.

The multicast flow is transitioned into named data MDT if the following conditions are met:

- The criteria for a flow to be part of the data MDT is met.
- The flow satisfies the rules that are associated with the route-policy.

Named Data MDT

Based on the configured threshold, when the multicast flow is chosen to be transitioned to data MDT, the flow is compared against the configured route-policy. If it satisfies the specified conditions, the Cisco IOS XR software creates a named data MDT and transitions the flow to the new named data MDT.

The transport-specific parameters that are required to build the distribution tree based on the underlay transport (like FEC for MLDP) are autogenerated. Subsequent flows that match the policy and are mapped to the same named data MDT, are multiplexed to the same multicast transport tree in the core.

Even if you are using only named data MDTs, and the number of named data MDTs that can be configured is independent of the number of data MDTs configured, you have to configure a nonzero data MDT value for creating named data MDTs.

Flow-Mapping Rules for Named Data MDT

When you use route-policy to create named data MDTs, the following rules may be applicable:

- You can map disparate multicast flows to the same named data-mdt.
- You can specify multiple named data MDTs in one policy.
- Only the flows that are mapped using the route-policy use a named data MDT.

  Flows which do not match the route-policy but are eligible to be transitioned to data-mdt, are transitioned to regular data-mdt.

- You can map a named-data MDT to only one path (Refer Chapter: Flexible Algorithm for MLDP).

  You can have a rule for creating a new named-data-mdt, that uses the same flexible algorithm which is used by another rule.

- You can map the named data MDT to only one color (configured using the **set on-demand-color <val>** command), when Tree-SID is used as core.

- Named data MDTs with the same name in different VRFs create different named data MDTs.
- If you use the same route-policies across VRFs, it creates different data MDTs.
- Named data MDTs with same name in IPv4 and IPv6 AF of a VRF creates different named data MDTs.
- Flows that do not match the route policy are mapped to dynamic MDTs.

Note Any change to the route-policy, deletes the existing named-data-mdt and recreates it.

Restrictions and Limitation of Named Data MDT

- The total number of named data MDTs and the number of dynamic data MDTs across all VRFs and address families (AF) must not exceed the router-supported limit for a given transport.
- The number of flows that can be transitioned to a single named data MDT is restricted to 255 by default. You can change this default 255 value using the **mdt data max-aggregation <value>** command. However, the total number of named data MDTs that can be created is not limited by the total number of regular data MDTs allowed by the configuration.

  When you update the maximum aggregation (**max-aggregation**) value, it does not re-evaluate the existing flows. However, the updated value is applicable to the new flows.

Configuring Named Data MDT

You can apply the route-policy for named data MDT using the following sample configurations and verify using the show commands:

```
Router(config)#multicast-routing vrf red address-family ipv4
Router(config-mcast-red-ipv4) #mdt data mldp 10 route-policy RedGroup-1
Router(config-mcast-red-ipv4) #
Router(config-mcast-red-ipv4) #exit

Router(config) #route-policy RedGroup-1
Router(config-rpl) #if destination in (228.0.0.0/24 le 32) then
Router(config-rpl-if) #set data-mdt RedGroup-1
Router(configq-rpl-if) #set flex-algo 128
Router(config-rpl-if) #pass
Router(config-rpl-if) #endif
Router(config-rpl) #end-policy
```

Verifying Named Data MDT

The following command outputs show the `Name` column which has the data MDT name.

```
Router#show pim vrt vonl mdt cache
Fri Aug 6 09:39:17.210 PDT

Core Source  Cust (Source, Group)     Core Data        Expires  Name
192.0.2.4  (31.3.233.7, 232.0.0.1)  [tree-id 524296]   never    RedGroup-1

  Leaf AD: 192.0.2.2
           192.0.2.1

192.0.2.4  (31.3.233.7, 232.0.0.2)  [tree-id 524296]    never    RedGroup-2
```

```
Leaf AD: 192.0.2.1


Router#show pim vrf vpn1 mdt sr-p2mp local
Fri Aug 6 09:39:20.435 PDT
Tree                        MDT          Cache  DIP Local  VRF Routes  On-demand Name
Identifier                  Source       Count      Entry  Using Cache Color
[tree-id 524296 (0x80008)]  192.0.2.4  2      N   Y      2           0           RedGroup-1

   Tree.SID Leaf:  192.0.2.1
                   192.0.2.2
Router#

Router#sh pim vrf vpn2 mdt cache
Tue Aug 17 04:39:58.751 PDT
Core Source   Cust (Source, Group)     Core Data      Expires Name
192.0.2.4   (31.3.234.7, 232.0.0.1)  [global-id 7]  00:02:14 n-mdt-vrf-vpn2
192.0.2.4   (31.3.234.7, 232.0.0.2)  [global-id 7]  00:02:14 n-mdt-vrf-vpn2
192.0.2.4   (31.3.234.7, 232.0.0.3)  [global-id 7]  00:02:14 n-mdt-vrf-vpn2
Router#
Router#

Router#sh pim vrf von2 mdt mldp local
Tue Aug 17 04:40:19.160 PDT
Core       MDT     Cache Max DIP Local VRF Routes    Name
Identifier Source  Count Agg     Entry Using Cache
[global-id 71      3     255 N   Y     3             n-mdt-vrf-von2
Router#
```

Internet Group Management Protocol

Cisco IOS XR Software provides support for Internet Group Management Protocol (IGMP) over IPv4.

IGMP provides a means for hosts to indicate which multicast traffic they are interested in and for routers to control and limit the flow of multicast traffic throughout the network. Routers build state by means of IGMP messages; that is, router queries and host reports.

A set of routers and hosts that receive multicast data streams from the same source is called a multicast group. Hosts use IGMP messages to join and leave multicast groups.

Note IGMP messages use group addresses, which are Class D IP addresses. The high-order four bits of a Class D address are 1110. Host group addresses can be in the range 224.0.0.0 to 239.255.255.255. The address is guaranteed not to be assigned to any group. The address 224.0.0.1 is assigned to all systems on a subnet. The address 224.0.0.2 is assigned to all routers on a subnet.

NCS 5500 supports IGMPv3 by default. No configuration is required. IGMP Version 3 permits joins and leaves for certain source and group pairs instead of requesting traffic from all sources in the multicast group.

Restrictions

IGMP snooping under VPLS bridge domain is not supported.

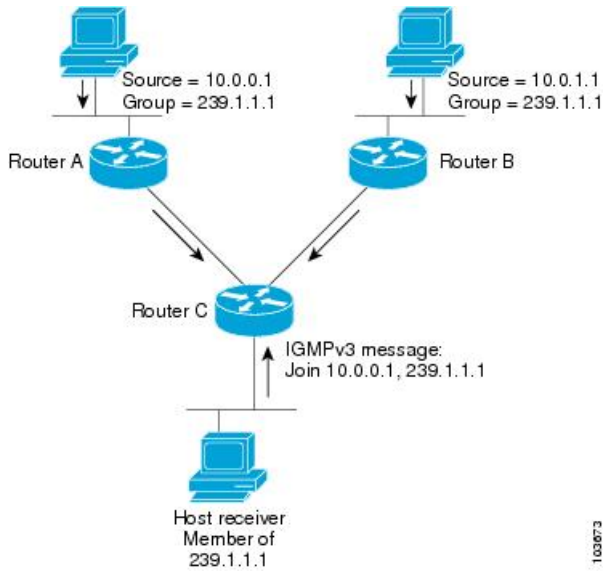
Functioning of IGMP Routing

The following image "IGMP Singaling" , illustrates two sources, 10.0.0.1 and 10.0.1.1, that are multicasting to group 239.1.1.1.

The receiver wants to receive traffic addressed to group 239.1.1.1 from source 10.0.0.1 but not from source 10.0.1.1.

The host must send an IGMPv3 message containing a list of sources and groups (S, G) that it wants to join and a list of sources and groups (S, G) that it wants to leave. Router C can now use this information to prune traffic from Source 10.0.1.1 so that only Source 10.0.0.1 traffic is being delivered to Router C.

Figure 4: IGMP Signaling



Configuring Maximum IGMP Per Interface Group Limit

The IGMP Per Interface States Limit sets a limit on creating OIF for the IGMP interface. When the set limit is reached, the group is not accounted against this interface but the group can exist in IGMP context for some other interface.

- If a user has configured a maximum of 20 groups and has reached the maximum number of groups, then no more groups can be created. If the user reduces the maximum number of groups to 10, the 20 joins will remain and a message of reaching the maximum is displayed. No more joins can be added until the number of groups has reached less than 10.
- If a user already has configured a maximum of 30 joins and add a max of 20, the configuration occurs displaying a message that the maximum has been reached. No state change occurs and also no more joins can occur until the threshold number of groups is brought down below the maximum number of groups.

Configuration Example

Configures all interfaces with 4000 maximum groups per interface except TenGigE interface 0/0/0/6, which is set to 3000:

```
Router#config
Router(config)#router igmp
Router(config-igmp)#maximum groups-per-interface 4000
Router(config-igmp)#interface TenGigE0/0/0/6
Router(config-igmp-default-if)#maximum groups-per-interface 3000
Router(config-igmp-default-if)#commit
```

Running Configuration

```
router igmp
 interface TenGigE0/0/0/6
  maximum groups-per-interface 3000
 !
 maximum groups-per-interface 4000
!
```

Verification

```
Router#show igmp summary
Robustness Value 2
No. of Group x Interfaces 37
Maximum number of Group x Interfaces 50000
Supported Interfaces   : 9
Unsupported Interfaces: 0
Enabled Interfaces     : 8
Disabled Interfaces    : 1
MTE tuple count        : 0
Interface                       Number  Max #
                                Groups  Groups
Loopback0                        4       4000
TenGigE0/0/0/0                   5       4000
TenGigE0/0/0/1                   5       4000
TenGigE0/0/0/2                   0       4000
TenGigE0/0/0/3                   5       4000
TenGigE0/0/0/6                   5       3000
TenGigE0/0/0/18                  5       4000
TenGigE0/0/0/19                  5       4000
TenGigE0/0/0/6.1                 3       4000
```

SSM Static Source Mapping

Configure a source (1.1.1.1) as part of a set of sources that map SSM groups described by the specified access-list (4).

Configuration Example

```
Router#configure
Router(config)#ipv4 access-list 4
Router(config-ipv4-acl)#permit ipv4 any 229.1.1.0 0.0.0.255
Router(config-ipv4-acl)#exit
Router(config)# multicast-routing
Router(config-mcast)#address-family ipv4
Router(config-mcast-default-ipv4)#ssm range 4
Router(config-mcast-default-ipv4)#exit
Router(config-mcast)#exit
Router(config)#router igmp
Router(config-igmp)#ssm map static 1.1.1.1 4
*/Repeat the above step as many times as you have source addresses to include in the set
for SSM mapping/*
Router(config-igmp)#interface TenGigE0/0/0/3
Router(config-igmp-default-if)#static-group 229.1.1.1
Router(config-igmp-default-if)#commit
```

Running Configuration

```
Router#show run multicast-routing
multicast-routing
 address-family ipv4
```

```
  ssm range 4
  interface all enable
 !
!
Router#show access-lists 4
ipv4 access-list 4
 10 permit ipv4 any 229.1.1.0 0.0.0.255

Router#show run router igmp
router igmp
 interface TenGigE0/0/0/3
 static-group 229.1.1.1
 !
 ssm map static 1.1.1.1 4
```

Verification

Verify if the parameters are set according to the configured values:

```
Router#show mrib route 229.1.1.1 detail
IP Multicast Routing Information Base
Entry flags: L - Domain-Local Source, E - External Source to the Domain,
    C - Directly-Connected Check, S - Signal, IA - Inherit Accept,
    IF - Inherit From, D - Drop, ME - MDT Encap, EID - Encap ID,
    MD - MDT Decap, MT - MDT Threshold Crossed, MH - MDT interface handle
    CD - Conditional Decap, MPLS - MPLS Decap, EX - Extranet
    MoFE - MoFRR Enabled, MoFS - MoFRR State, MoFP - MoFRR Primary
    MoFB - MoFRR Backup, RPFID - RPF ID Set, X - VXLAN
Interface flags: F - Forward, A - Accept, IC - Internal Copy,
    NS - Negate Signal, DP - Don't Preserve, SP - Signal Present,
    II - Internal Interest, ID - Internal Disinterest, LI - Local Interest,
    LD - Local Disinterest, DI - Decapsulation Interface
    EI - Encapsulation Interface, MI - MDT Interface, LVIF - MPLS Encap,
    EX - Extranet, A2 - Secondary Accept, MT - MDT Threshold Crossed,
    MA - Data MDT Assigned, LMI - mLDP MDT Interface, TMI - P2MP-TE MDT Interface
    IRMI - IR MDT Interface
(1.1.1.1,229.1.1.1) RPF nbr: 1.1.1.1 Flags: RPF
  Up: 00:01:11
  Incoming Interface List
    Loopback0 Flags: A, Up: 00:01:11
  Outgoing Interface List
    TenGigE0/0/0/3 Flags: F NS LI, Up: 00:01:11
```

Information About IGMP Snooping Configuration Profiles

To enable IGMP snooping on a bridge domain, you must attach a profile to the bridge domain. The minimum configuration is an empty profile. An empty profile enables the default configuration options and settings for IGMP snooping, as listed in the *Default IGMP Snooping Configuration Settings.*

You can attach IGMP snooping profiles to bridge domains or to ports under a bridge domain. The following guidelines explain the relationships between profiles attached to ports and bridge domains:

- Any IGMP profile attached to a bridge domain, even an empty profile, enables IGMP snooping. To disable IGMP snooping, detach the profile from the bridge domain.
- An empty profile configures IGMP snooping on the bridge domain and all ports under the bridge using default configuration settings.
- A bridge domain can have only one IGMP snooping profile attached to it (at the bridge domain level) at any time. Profiles can be attached to ports under the bridge, one profile per port.

- Port profiles are not in effect if the bridge domain does not have a profile attached to it.
- IGMP snooping must be enabled on the bridge domain for any port-specific configurations to be in effect.
- If a profile attached to a bridge domain contains port-specific configuration options, the values apply to all of the ports under the bridge, including all mrouter and host ports, unless another port-specific profile is attached to a port.
- When a profile is attached to a port, IGMP snooping reconfigures that port, disregarding any port configurations that may exist in the bridge-level profile.

Note

- IEEE 802.1Q, Q-in-Q, default and untagged encapsulations are supported.
- IEEE 802.1ad and other encapsulations are not supported.
- IEEE 802.1ad is not supported with BVI.

Creating Profiles

To create a profile, use the **igmp snooping profile** command in global configuration mode.

Attaching and Detaching Profiles

To attach a profile to a bridge domain, use the **mldp snooping profile** command in l2vpn bridge group bridge domain configuration mode. To attach a profile to a port, use the **mldp snooping profile** command in the interface configuration mode under the bridge domain. To detach a profile, use the **no** form of the command in the appropriate configuration mode.

When you detach a profile from a bridge domain or a port, the profile still exists and is available for use at a later time. Detaching a profile has the following results:

- If you detach a profile from a bridge domain, MLDP snooping is deactivated in the bridge domain.
- If you detach a profile from a port, MLDP snooping configuration values for the port are instantiated from the bridge domain profile.

Changing Profiles

You cannot make changes to an active profile. An active profile is one that is currently attached.

If you need to change an active profile, you must detach it from all bridges or ports, change it, and reattach it.

Another way to do this is to create a new profile incorporating the desired changes and attach it to the bridges or ports, replacing the existing profile. This deactivates IGMP snooping and then reactivates it with parameters from the new profile.

Configuring Access Control

Access control configuration is the configuration of access groups and weighted group limits.

The role of access groups in IGMP v2/v3 message filteringis to permit or deny host membership requests for multicast groups (*,G) and multicast source groups (S,G). This is required to provide blocked and allowed list access to IPTV channel packages.

Weighted group limits restrict the number of IGMP v2/v3 groups, in which the maximum number of concurrently allowed multicast channels can be configured on a per EFP- and per PW-basis.

IGMP Snooping Access Groups

Although Layer-3 IGMP routing also uses the **igmp access-group** command in support of access groups, the support is not the same in Layer-2 IGMP, because the Layer-3 IGMP routing access group feature does not support source groups.

Access groups are specified using an extended IP access list referenced in an IGMP snooping profile that you attach to a bridge domain or a port.

Note A port-level access group overrides any bridge domain-level access group.

The **access-group** command instructs IGMP snooping to apply the specified access list filter to received membership reports. By default, no access list is applied.

Changes made to the access-list referenced in the profile (or a replacement of the access-list referenced in the igmp snooping profile) will immediately result in filtering the incoming igmp group reports and the existing group states accordingly, without the need for a detach-reattach of the igmp snooping profile in the bridge-domain, each time such a change is made.

IGMP Snooping Group Weighting

To limit the number of IGMP v2/v3 groups, in which the maximum number of concurrently allowed multicast channels must be configurable on a per EFP-basis and per PW-basis, configure group weighting.

IGMP snooping limits the membership on a bridge port to a configured maximum, but extends the feature to support IGMPv3 source groups and to allow different weights to be assigned to individual groups or source groups. This enables the IPTV provider, for example, to associate standard and high- definition IPTV streams, as appropriate, to specific subscribers.

This feature does not limit the actual multicast bandwidth that may be transmitted on a port. Rather, it limits the number of IGMP groups and source-groups, of which a port can be a member. It is the responsibility of the IPTV operator to configure subscriber membership requests to the appropriate multicast flows.

The **group policy** command, which is under igmp-snooping-profile configuration mode, instructs IGMP snooping to use the specified route policy to determine the weight contributed by a new <*,G> or <S,G> membership request. The default behavior is for there to be no group weight configured.

The **group limit** command specifies the group limit of the port. No new group or source group is accepted if its contributed weight would cause this limit to be exceeded. If a group limit is configured (without group policy configuration), a <S/*,G> group state will have a default weight of 1 attributed to it.

Note By default, each group or source-group contributes a weight of 1 towards the group limit. Different weights can be assigned to groups or source groups using the group policy command.

The group limit policy configuration is based on these conditions:

- Group weight values for <*,G> and <S,G> membership are configured in a Route Policy, that is included in an igmp snooping profile attached to a BD or port.
- Port level weight policy overrides any bridge domain level policy, if group-limit is set and route-policy is configured.
- If there is no policy configured, each group weight is counted equally and is equal to 1.
- If policy has been configured, all matching groups get weight of 1 and un-matched groups have 0 weight.

IPv6 Multicast for Multiple Sources

Table 3: Feature History Table

Feature Name	Release Information	Feature Description
IPv6 Multicast for Multiple Sources	Release 7.5.1	This feature is now supported on routers that have the Cisco NC57 line cards installed and operate in native and compatible modes. IPv6 multicast supports multiple sources for a single multicast group.

Note When a router has LCs (with and without external TCAMs), it operates with default IPv6 multicast route scale, which is programmed on the LC without an external TCAM.

Statistics for Ingress Multicast Routes

Multicast and interface statistics are often used for accounting purpose. By default Multicast Forwarding Information Base (MFIB) does not store multicast route statistics.

Note The MFIB counter can store a maximum of 2000 multicast routes, if the count exceeds beyond 2000 routes, then statistics are overwritten.

This table lists commands used to display or reset multicast route statistics stored in MFIB.

Table 4: show and clear commands for Multicast Statistics

Command	Description
show mfib hardware route <source-address> location <node-id>	Displays platform-specific MFIB information for the packet and byte counters for multicast routes originating from the specified source.

Command	Description
Show mfib hardware route <source-address><group-address> location <node-id>	Displays platform-specific MFIB information for the packet and byte counters for multicast routes originating from the specified source and belonging to specified multicast group.
clear mfib hardware ingress route statistics location <node-id>	Resets allocated counter values regardless of the MFIB hardware statistics mode from the designated node.
clear mfib hardware route <source-address> location <node-id>	Resets allocated counter values regardless of the MFIB hardware statistics mode from the designated node for specified multicast route source.
clear mfib hardware route <source-address><group-address> location <node-id>	Resets allocated counter values regardless of the MFIB hardware statistics mode from the designated node for specified multicast route source and multicast group.

Note To program IPv6 multicast routes in external TCAM, use the following commands:

- **hw-module profile tcam fib v6mcast**
- **hw-module profile tcam fib ipv6 unicast**

Configuring only **hw-module profile tcam fib v6mcast percent** *value* causes an unexpected behaviour for 5501-SE.

Configuring Statistics for Ingress Multicast Routes

Configuration Example

In this example you will enable MRIB route statistics logging for ingress multicast routes for all locations:

```
Router#config
Router(config)#hw-module profile mfib statistics
Router(config)#commit
Router(config)#exit
Router#admin
Router(admin)#reload location all      /*Reloads all line cards. This is required step,
else the multicast route statistcis will not be created.*/
```

Running Configuration

```
Router#show running config
hw-module profile mfib statistics
!
```

Verification

The below show commands display the multicast statistics for source (192.0.2.2), group (226.1.2.1) and node location 0/1/cpu0 for ingress route:

Note The multicast egress statistics per flow (per SG) is not suported. But egress interface level multicast statistics is supported. Also the drop statistics is not supported.

```
Router#show mfib hardware route statistics location 0/1/cpu0
(192.0.2.2, 226.1.2.1) :: Packet Stats 109125794 Byte Stats 23680297298
(192.0.2.2, 226.1.2.2) :: Packet Stats 109125760 Byte Stats 23680289920
(192.0.2.2, 226.1.2.3) :: Packet Stats 109125722 Byte Stats 23680281674
(192.0.2.2, 226.1.2.4) :: Packet Stats 109125683 Byte Stats 23680273211
(192.0.2.2, 226.1.2.5) :: Packet Stats 109125644 Byte Stats 23680264748
(192.0.2.2, 226.1.2.6) :: Packet Stats 109129505 Byte Stats 23681102585
(192.0.2.2, 226.1.2.7) :: Packet Stats 109129470 Byte Stats 23681094990
(192.0.2.2, 226.1.2.8) :: Packet Stats 109129428 Byte Stats 23681085876
(192.0.2.2, 226.1.2.9) :: Packet Stats 109129385 Byte Stats 23681076545
(192.0.2.2, 226.1.2.10) :: Packet Stats 109129336 Byte Stats 23681065912


Router#show mfib hardware route statistics 192.0.2.2 location 0/1/cpu0
(192.0.2.2, 226.1.2.1) :: Packet Stats 109184295 Byte Stats 23692992015
(192.0.2.2, 226.1.2.2) :: Packet Stats 109184261 Byte Stats 23692984637
(192.0.2.2, 226.1.2.3) :: Packet Stats 109184223 Byte Stats 23692976391
(192.0.2.2, 226.1.2.4) :: Packet Stats 109184184 Byte Stats 23692967928
(192.0.2.2, 226.1.2.5) :: Packet Stats 109184145 Byte Stats 23692959465
(192.0.2.2, 226.1.2.6) :: Packet Stats 109184106 Byte Stats 23692951002
(192.0.2.2, 226.1.2.7) :: Packet Stats 109184071 Byte Stats 23692943407
(192.0.2.2, 226.1.2.8) :: Packet Stats 109184029 Byte Stats 23692934293
(192.0.2.2, 226.1.2.9) :: Packet Stats 109183986 Byte Stats 23692924962


Router#show mfib hardware route statistics 192.0.2.2 226.1.2.1 location 0/1/cpu0
(192.0.2.2, 226.1.2.1) :: Packet Stats 109207695 Byte Stats 23698069815
```

Multicast Route Statistics

Table 5: Feature History Table

Feature Name	Release Information	Feature Description
Enhancement to Multicast Route Statistics	Release 7.3.1	When enabled, this feature provides information on the rate of packets received and OLE for a multicast route. Starting this release, the feature is extended on the Cisco NCS 5500 series routers. In addition, this feature is supported on ingress IPv6 stats on routers installed with SE cards.

Multicast route statistic feature provides information about the multicast routes. The multicast statistics information includes the rate at which packets are received.

Before enabling multicast route statistics, you must configure an ACL to specify which of the IP route statistics to be captured.

Restrictions for Implementing Multicast Route Statistics Feature

These are the points that you should consider before implementing multicast route statistics feature:

- Multicast route statistics are available for <S,G> routes only. The statistics for <*,G> routes are not available.
- IPv6 multicast route statistics are not supported.
- Multicast route statistics for egress direction is not supported.
- When ACL is mapped with **hw-module router-stats** configuration, you can't modify the ACL. To modify ACLs that are mapped with router-stats, remove the existing **hw-module router-stats** configuration and update the ACL entries. Then, configure the **hw-module router-stats** again.

This feature supports only:

- L3 Multicast traffic.

Configure Multicast Route Statistics

Table 6: Feature History Table

Feature Name	Release Information	Description
YANG Data Models for Multicast Interface Counters	Release 7.4.1	This feature introduces YANG data model support for multicast packets, in and out bytes per interface and sub-interface. With this feature, you can programmatically retrieve the operational details of multicast interfaces. You can access the data models from the Github repository.

Configuring multicast route statistics includes these main tasks:

- Configuring an ACL
- Enabling multicast route statistics for the configured ACLs

```
RP0/0/RP0/CPU0:router# configure

/* Configure an ACL matching the (S,G) routes for which statistics have to be captured:*/
RP0/0/RP0/CPU0:router(config)# ipv4 access-list mcast-counter
RP0/0/RP0/CPU0:router(config-acl)# 10 permit ipv4 host 10.1.1.2 host 224.2.151.1
RP0/0/RP0/CPU0:router(config-acl)# 30 permit ipv4 10.1.1.0/24 232.0.4.0/22
RP0/0/RP0/CPU0:router(config-acl)# 50 permit ipv4 192.168.0.0/24 232.0.4.0/22
RP0/0/RP0/CPU0:router(config-acl)#commit
RP0/0/RP0/CPU0:router(config-acl)#exit

/* Enable multicast route statistics for the configured ACL in the ingress direction on the
```

```
 default VRF. */
RP0/0/RP0/CPU0:router(config)# hw-module route-stats l3mcast vrf default ipv4 egress
mcast-router
RP0/0/RP0/CPU0:router(config)# hw-module route-stats l3mcast vrf default ipv4 ingress
mcast-router
```

Note

- If you are enabling the route stats for a router on the global table, use **vrf default**. If you are enabling the route stats for specific vrf, use the **vrf** *vrfname* option.
- In case, you want to enable route stats for all tables, do not use the **vrf**.

  For example:

  ```
  RP0/0/RP0/CPU0:router(config)#hw-module route-stats l3mcast ipv4 ingress mcast-counter
  ```

- If you configure **hw-module route-stats** on both vrf default and vpn routes for either IPv4 or IPv6 ACLs, then to switch vrf default to other VRF, remove the configuration of the existing **hw-module route-stats** and commit it, and then configure the hw-module stats with the required vrf and commit it.

Verification

Use the **show mfib route rate** command to verify if the multicast route information is captured for the traffic that matches the ACL:

Note The ingress stats are always per S, G.

```
RP0/0/RP0/CPU0:router# show mfib route rate
Thu Aug 16 18:04:47.312 PDT

IP Multicast Forwarding Rates
(Source Address, Group Address)
     Incoming rate:
           Node: (Incoming node) : pps/bps
     Outgoing rate:
           Node: (Outgoing node) : pps/bps

(10.1.1.2,232.0.0.1)
     Incoming rate :
           Node : 0/0/CPU0 : 4593 / 18153671
     Outgoing rate :
           Node : 0/0/CPU0 : 0 / 0
```

The above output shows that the multicast source **10.1.1.2** is sending packets to multicast group **232.0.0.1** and is received at **4593** pps.

IPv6 Egress Multicast Route Statistics Example

```
RP0/0/RP0/CPU0:router# configure

/* Configure an ACL matching the (S,G) routes for which statistics have to be captured:*/
RP0/0/RP0/CPU0:router(config)# ipv6 access-list 12
RP0/0/RP0/CPU0:router(config-acl)# 10 permit ipv6 any ff33:1:3::1/48
RP0/0/RP0/CPU0:router(config-acl)#commit
```

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

/* Enable multicast route statistics for the configured ACL in the egress direction on the
 named VRF. */
RP0/0/RP0/CPU0:router(config)# hw-module route-stats l3mcast vrf vrf1 ipv6 egress 12
```

For the information on the interface accounting stats, use the show interface accounting command. The following show command displays interface accounting stats for ingress:

```
Router# show int tenGigE 0/0/0/15 accounting
Mon Nov 12 10:26:20.592 UTC
TenGigE0/0/0/15
  Protocol              Pkts In         Chars In     Pkts Out        Chars Out
  IPV6_MULTICAST    22125711958    1814308380556            0                0
  IPV6_ND                     0                0         1243           128960
```

Cisco IOS XR Release 7.4.1 and later support YANG data model for multicast interface counters.

- Cisco-IOS-XR-infra-statsd-oper:infra-statistics/interfaces/interface/protocols/protocol
- Cisco-IOS-XR-infra-statsd-oper:infra-statistics/interfaces/interface[interface-name=TenGigE0/0/0/18]/protocols/protocol

Note The YANG model does not support ingress and egress multicast route stats.

The following show command displays interface accounting stats for egress:

```
Router# show interfaces bundle-ether 100.1001 accounting rates
Mon Aug 26 15:56:41.738 IST
Bundle-Ether100.1001
                              Ingress                        Egress
  Protocol            Bits/sec         Pkts/sec     Bits/sec         Pkts/sec
  IPV4_MULTICAST             0                0     11455000              990
  IPV6_MULTICAST             0                0     11455000              990
  ARP                        0                0            0                0
  IPV6_ND                    0                0            0                0
```

Bundle Member Selection

Table 7: Feature History Table

Feature Name	Release Information	Feature Description
Bundle Member Selection	Release 7.3.1	This feature enables selecting a bundle member in the control plane to steer the L2 and L3 multicast traffic traversing over bundle at the egress NP. This feature helps optimize fabric bandwidth as the member selection is performed in the control plane.

This feature enables selecting a bundle member in the control plane to steer the L2 and L3 multicast traffic traversing over bundle at the egress NP.

This feature brings following benefits:

- Optimizes fabric bandwidth as the member selection is performed in the control plane
- Reduces NP bandwidth and processing as number of OLE replications are less
- Supports bundle member change in MVPN head node with local receiver

Multicast Over IPV4 Unicast GRE Tunnels

Table 8: Feature History Table

Feature Name	Release Information	Feature Description
Multicast Over IPV4 Unicast GRE Tunnels	Release 7.5.1	This feature is now supported on routers that have the Cisco NC57 line cards installed and operate in native and compatible modes. Multicast over GRE allows encapsulation of multicast packets using GRE tunnels, thereby enabling transport of multicast packets securely between source and destination routers located in different IP clouds.
Support for Multicast Over IPV4 Unicast GRE Tunnels	Release 7.2.2	This feature allows encapsulation of multicast packets using GRE tunnels, thereby enabling transport of multicast packets securely between source and destination routers located in different IP clouds.

Use IPV4 unicast GRE tunnels to transport multicast traffic securely over the network.

Generic Routing Encapsulation (GRE) is a tunneling protocol that encapsulates and transport packets of one protocol over another protocol.

If you want to send multicast packets from a source to destination router configured with a different routing protocol, you can encapsulate the packets using GRE unicast tunnels. The encapsulated packets are forwarded like any other IPv4 unicast packet to the destination endpoint of the tunnel. The destination router then de-encapsulates the packet to retrieve the multicast packets.

For more information Configuring GRE Tunnels.

Restrictions

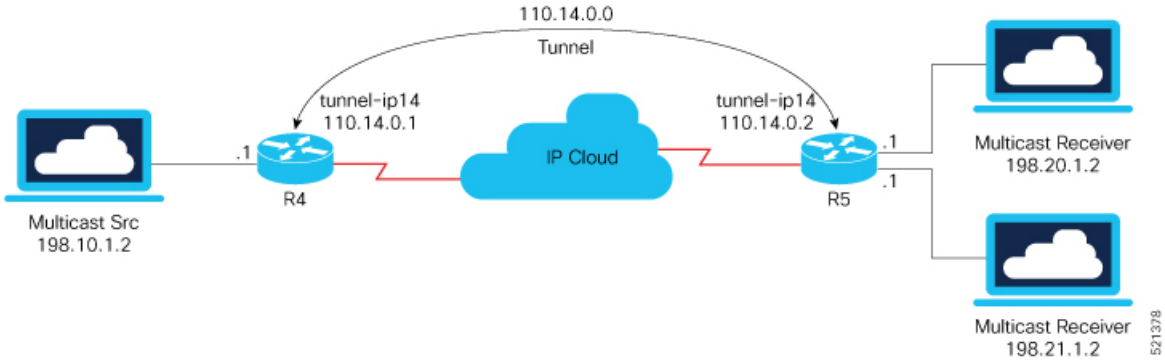
- ECMP and LAG hash based on GRE header is not supported on the NC55 line cards.
- Multicast over GRE with L3VPN is only supported on UFI and not on the NC55 and NC57 line cards.
- Supported only on Cisco NCS 5501-SE routers.

- Only SSM V4 and V6 address family traffic is supported.
- Supports up to 500 GRE tunnels
- Only up to 16 unique source IP addresses are supported for the tunnel source
- Multicast over GRE tunnel is supported only with 2-pass GRE tunnel configuration.
- Configurable MTU is not supported on Single-pass GRE interface, but supported on 2-pass GRE interface.
- This is a native Multicast over GRE feature and not Multicast VPN (mVPN) Profile 0 or Rosen GRE.

Configuration

In this topology, the multicast source (198.10.1.2) is connected to R4. The multicast receivers are connected to R5 and is configured to receive multicast packets. Separating the source (R4) and receiver (R5) is an IP cloud, which is not configured for multicast routing.

Multicast packets are encapsulated with GRE headers and transported via GRE tunnel (tunnel-ip14).



Configuration Example

```
R4

interface TenGigE0/0/0/32.1
 ipv4 address 198.10.1.1 255.255.255.0
 ipv6 address 2002:10:1::1/64
 encapsulation dot1q 1
!

interface Loopback14
 ipv4 address 10.10.10.14 255.255.255.255
!
interface tunnel-ip14
 ipv4 address 110.14.0.1 255.255.255.0
 ipv6 address 110:14::1/64
 tunnel mode gre ipv4
 tunnel source 10.10.10.14
 tunnel destination 20.20.20.14
!
router ospf core_native_mcast
 nsr
 router-id 10.10.10.1
 area 0.0.0.0
  interface Bundle-Ether121
  !
```

```
  interface Loopback14
  !

router ospf mogre_edge_native_mcast
 nsr
 router-id 10.10.10.11
 area 0.0.0.0
  interface tunnel-ip14
  !
  interface TenGigE0/0/0/32.1


R5

interface TenGigE0/0/0/0.1
 ipv4 address 198.20.1.1 255.255.255.0
 ipv6 address 2002:20:1::1/64
 encapsulation dot1q 1
!


interface TenGigE0/0/0/1.1
 ipv4 address 198.21.1.1 255.255.255.0
 ipv6 address 2002:21:1::1/64
 encapsulation dot1q 1
!

interface Loopback14
 ipv4 address 20.20.20.14 255.255.255.255
!

interface tunnel-ip14
 ipv4 address 110.14.0.2 255.255.255.0
 ipv6 address 110:14::2/64
 tunnel mode gre ipv4
 tunnel source 20.20.20.14
 tunnel destination 10.10.10.14
!


router ospf core_native_mcast
 nsr
 router-id 20.20.20.1
 area 0.0.0.0
  interface Bundle-Ether121
  !
  interface Loopback14


router ospf mogre_edge_native_mcast
 nsr
 router-id 20.20.20.11
 area 0.0.0.0
  interface tunnel-ip14
  !
  interface TenGigE0/0/0/0.1
  !
  interface TenGigE0/0/0/1.1
```

Verification

In this example, Router R4 receives joins for group 232.1.0.0 from source 198.10.1.2 connected to R4. PIM joins are traversed from R5 to R4, as shown in the **show PIM topology** command output. The joins are learnt on Router R4 via tunnel-ip, as it acts as the transport layer.

```
RP/0/RP0/CPU0:R4# show pim topology 232.1.0.0 198.10.1.2

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, IA - Inherit Alive, LH - Last Hop
    DSS - Don't Signal Sources,  RR - Register Received
    SR - Sending Registers, SNR - Sending Null Registers
    E - MSDP External, EX - Extranet
    MFA - Mofrr Active, MFP - Mofrr Primary, MFB - Mofrr Backup
    DCC - Don't Check Connected, ME - MDT Encap, MD - MDT Decap
    MT - Crossed Data MDT threshold, MA - Data MDT Assigned
    SAJ - BGP Source Active Joined, SAR - BGP Source Active Received,
    SAS - BGP Source Active Sent, IM - Inband mLDP, X - VxLAN
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, EX - Extranet,
    BGP - BGP C-Multicast Join, BP - BGP Source Active Prune,
    MVS - MVPN Safi Learned, MV6S - MVPN IPv6 Safi Learned

(198.10.1.2,232.1.0.0)SPT SSM Up: 00:33:51
JP: Join(now) RPF: TenGigE0/0/0/32.1,198.10.1.2* Flags:
  tunnel-ip14                 00:29:34  fwd Join(00:03:00)
```

From the following **show mrib route** command output, you can see that TenGigE0/0/0/32.1 is the incoming interface and tunnel-ip14 is the outgoing interface for (S,G) 198.10.1.2, 232.1.0.0.

```
RP/0/RP0/CPU0:R4# show mrib route 232.1.0.0 detail
IP Multicast Routing Information Base
Entry flags: L - Domain-Local Source, E - External Source to the Domain,
    C - Directly-Connected Check, S - Signal, IA - Inherit Accept,
    IF - Inherit From, D - Drop, ME - MDT Encap, EID - Encap ID,
    MD - MDT Decap, MT - MDT Threshold Crossed, MH - MDT interface handle
    CD - Conditional Decap, MPLS - MPLS Decap, EX - Extranet
    MoFE - MoFRR Enabled, MoFS - MoFRR State, MoFP - MoFRR Primary
    MoFB - MoFRR Backup, RPFID - RPF ID Set, X - VXLAN
Interface flags: F - Forward, A - Accept, IC - Internal Copy,
    NS - Negate Signal, DP - Don't Preserve, SP - Signal Present,
    II - Internal Interest, ID - Internal Disinterest, LI - Local Interest,
    LD - Local Disinterest, DI - Decapsulation Interface
    EI - Encapsulation Interface, MI - MDT Interface, LVIF - MPLS Encap,
    EX - Extranet, A2 - Secondary Accept, MT - MDT Threshold Crossed,
    MA - Data MDT Assigned, LMI - mLDP MDT Interface, TMI - P2MP-TE MDT Interface
    IRMI - IR MDT Interface, TRMI - TREE SID MDT Interface, MH - Multihome Interface

(198.10.1.2,232.1.0.0) Ver: 0x6e42 RPF nbr: 198.10.1.2 Flags: RPF, FGID: 16915, Statistics
 enabled: 0x0, Tunnel RIF: -1
  Up: 00:32:54
 Incoming Interface List
    TenGigE0/0/0/32.1 Flags: F A LI, Up: 00:32:54
  Outgoing Interface List
    tunnel-ip14 (0/0/0) Flags: F NS, Up: 00:28:37

RP/0/RP0/CPU0:R4#sh mfib route 232.1.0.0 198.10.1.2

IP Multicast Forwarding Information Base
Entry flags: C - Directly-Connected Check, S - Signal, D - Drop,
```

```
  IA - Inherit Accept, IF - Inherit From, EID - Encap ID,
  ME - MDT Encap, MD - MDT Decap, MT - MDT Threshold Crossed,
  MH - MDT interface handle, CD - Conditional Decap,
  DT - MDT Decap True, EX - Extranet, RPFID - RPF ID Set,
  MoFE - MoFRR Enabled, MoFS - MoFRR State, X - VXLAN
Interface flags: F - Forward, A - Accept, IC - Internal Copy,
  NS - Negate Signal, DP - Don't Preserve, SP - Signal Present,
  EG - Egress, EI - Encapsulation Interface, MI - MDT Interface,
  EX - Extranet, A2 - Secondary Accept
Forwarding/Replication Counts: Packets in/Packets out/Bytes out
Failure Counts: RPF / TTL / Empty Olist / Encap RL / Other

(198.10.1.2,232.1.0.0),  Flags:
  Up: 00:35:39
  Last Used: never
  SW Forwarding Counts: 0/0/0
  SW Replication Counts: 0/0/0
  SW Failure Counts: 0/0/0/0/0
 tunnel-ip14 (0xe0) Flags:  NS, Up:00:31:16


RP/0/RP0/CPU0:R4# show mfib hardware route 232.1.0.0 198.10.1.2 location 0/0/cPU0
Route (198.10.1.2: 232.1.0.0)
   HAL PD context
       VRF ID: 0 Core MCID : 0 Core backup MCID 0

   HAL Ingress route context:
       Route FGID: 16915 RPF IF signal : not-set Local receivers: set
       Encap ID flag: not-set, Encap ID: 0
       Tunnel RIF: 0x0
       Statistics enabled: not-set

       Ingress engine context:
          local_route: set, is_accept_intf_bvi: not-set is_tun_rif_set:not-set
           VRF ID: 0 RPF ID:0 Tunnel RIF:0x0
   HAL Egress route context:
       RPF ID: 0

       Egress engine context:
          out_of_sync: not-set, local_intf: not-set
           bvi_count: 0

   DPA Route context:
      Handle: 30895ef540
      Transaction ID: 91864
      Number of OLE: 4   VRF ID: 0
      Incoming interface : Te0/0/0/32.1 A_intf_id: 0x39 Merged flag 0
      Tunnel RIF : 0x0 FGID: 16915
      FEC ID : 0x2001f888 Punt action: 0x0
      TCAM entry ID : 0x0 IPMC action: 0x4 FEC Accessed 1
      L3 Intf Refhandle : 0x308ccbd448 L3 interface ref key: 0x0
      Statistics enabled : not-set Statistics activated : not-set

      Egress Route OLEs:
        Handle: 308e669960
        Transaction ID: 267707
        NPU ID: 0 Outgoing intf: ti14
        OLE Type : Gre tunnel interface
        outgoing port : 0x0 cud: 0x13878 is_bundle: 0
        Sys_port : 0x0 mpls encap id: 0x0 LAG ID: 0
        is_pw_access: 0 pw_encap_id:0
        L3 intf refhndl : 0x308d47bee8 L3 intf refkey: 0x200040fc
        L2 Port refhandle : 0x0 L2 Port refkey: 0x0
        MPLS nh refhandle : 0x0 MPLS nh refkey: 0x0
```

```
        LAG port refhandle : 0x0 LAG port refkey: 0x0
        EFP-Visibility: not-set
        Total fwd packets : 0 Total fwd bytes: 0
```

The following command output shows that the PIM joins received from TenGigE0/0/0/0.1 and TenGigE0/0/0/1.1 for group 232.1.0.0.

```
RP/0/RP0/CPU0:R5# show pim topology 232.1.0.0 198.10.1.2

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, IA - Inherit Alive, LH - Last Hop
    DSS - Don't Signal Sources,  RR - Register Received
    SR - Sending Registers, SNR - Sending Null Registers
    E - MSDP External, EX - Extranet
    MFA - Mofrr Active, MFP - Mofrr Primary, MFB - Mofrr Backup
    DCC - Don't Check Connected, ME - MDT Encap, MD - MDT Decap
    MT - Crossed Data MDT threshold, MA - Data MDT Assigned
    SAJ - BGP Source Active Joined, SAR - BGP Source Active Received,
    SAS - BGP Source Active Sent, IM - Inband mLDP, X - VxLAN
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, EX - Extranet,
    BGP - BGP C-Multicast Join, BP - BGP Source Active Prune,
    MVS - MVPN Safi Learned, MV6S - MVPN IPv6 Safi Learned

(198.10.1.2,232.1.0.0)SPT SSM Up: 00:44:09
JP: Join(00:00:02) RPF: tunnel-ip14,110.14.0.1 Flags:
  TenGigE0/0/0/0.1            00:44:09  fwd LI LH
  TenGigE0/0/0/1.1            00:44:06  fwd LI LH
```

The traffic received from `tunnel-ip` is then forwarded to multicast receivers TenGigE0/0/0/0.1 and TenGigE0/0/0/1.1 acting as OLE, as shown in the following output.

```
RP/0/RP0/CPU0:R5# show mrib route 232.1.0.0 detail
IP Multicast Routing Information Base
Entry flags: L - Domain-Local Source, E - External Source to the Domain,
    C - Directly-Connected Check, S - Signal, IA - Inherit Accept,
    IF - Inherit From, D - Drop, ME - MDT Encap, EID - Encap ID,
    MD - MDT Decap, MT - MDT Threshold Crossed, MH - MDT interface handle
    CD - Conditional Decap, MPLS - MPLS Decap, EX - Extranet
    MoFE - MoFRR Enabled, MoFS - MoFRR State, MoFP - MoFRR Primary
    MoFB - MoFRR Backup, RPFID - RPF ID Set, X - VXLAN
Interface flags: F - Forward, A - Accept, IC - Internal Copy,
    NS - Negate Signal, DP - Don't Preserve, SP - Signal Present,
    II - Internal Interest, ID - Internal Disinterest, LI - Local Interest,
    LD - Local Disinterest, DI - Decapsulation Interface
    EI - Encapsulation Interface, MI - MDT Interface, LVIF - MPLS Encap,
    EX - Extranet, A2 - Secondary Accept, MT - MDT Threshold Crossed,
    MA - Data MDT Assigned, LMI - mLDP MDT Interface, TMI - P2MP-TE MDT Interface
    IRMI - IR MDT Interface, TRMI - TREE SID MDT Interface, MH - Multihome Interface

(198.10.1.2,232.1.0.0) Ver: 0x8ef6 RPF nbr: 110.14.0.1 Flags: RPF, FGID: 28465, Statistics
 enabled: 0x0, Tunnel RIF: -1
  Up: 00:42:42
  Incoming Interface List
    tunnel-ip14 Flags: A, Up: 00:38:24
  Outgoing Interface List
    TenGigE0/0/0/0.1 Flags: F NS LI, Up: 00:42:42
```

```
    TenGigE0/0/0/1.1 Flags: F NS LI, Up: 00:42:40

RP/0/RP0/CPU0:R5# show mfib route 232.1.0.0 198.10.1.2

IP Multicast Forwarding Information Base
Entry flags: C - Directly-Connected Check, S - Signal, D - Drop,
  IA - Inherit Accept, IF - Inherit From, EID - Encap ID,
  ME - MDT Encap, MD - MDT Decap, MT - MDT Threshold Crossed,
  MH - MDT interface handle, CD - Conditional Decap,
  DT - MDT Decap True, EX - Extranet, RPFID - RPF ID Set,
  MoFE - MoFRR Enabled, MoFS - MoFRR State, X - VXLAN
Interface flags: F - Forward, A - Accept, IC - Internal Copy,
  NS - Negate Signal, DP - Don't Preserve, SP - Signal Present,
  EG - Egress, EI - Encapsulation Interface, MI - MDT Interface,
  EX - Extranet, A2 - Secondary Accept
Forwarding/Replication Counts: Packets in/Packets out/Bytes out
Failure Counts: RPF / TTL / Empty Olist / Encap RL / Other

(198.10.1.2,232.1.0.0),  Flags:
  Up: 00:43:30
  Last Used: never
  SW Forwarding Counts: 0/0/0
  SW Replication Counts: 0/0/0
  SW Failure Counts: 0/0/0/0/0
  tunnel-ip14 Flags:  A, Up:00:39:09
  TenGigE0/0/0/0.1 Flags:  NS, Up:00:43:30
  TenGigE0/0/0/1.1 Flags:  NS, Up:00:43:17


RP/0/RP0/CPU0:R5# show mfib hardware route 232.1.0.0 198.10.1.2 location 0/0/CPU0

Route (198.10.1.2: 232.1.0.0)
   HAL PD context
       VRF ID: 0 Core MCID : 0 Core backup MCID 0

   HAL Ingress route context:
       Route FGID: 28465 RPF IF signal : not-set Local receivers: set
       Encap ID flag: not-set, Encap ID: 0
       Tunnel RIF: 0x0
       Statistics enabled: not-set

       Ingress engine context:
          local_route: set, is_accept_intf_bvi: not-set is_tun_rif_set:not-set
           VRF ID: 0 RPF ID:0 Tunnel RIF:0x0
   HAL Egress route context:
       RPF ID: 0

       Egress engine context:
          out_of_sync: not-set, local_intf: not-set
           bvi_count: 0

   DPA Route context:
      Handle: 308852aed0
      Transaction ID: 228831
      Number of OLE: 2   VRF ID: 0
      Incoming interface : ti14 A_intf_id: 0x43 Merged flag 0
      Tunnel RIF : 0x0 FGID: 28465
      FEC ID : 0x2001fd37 Punt action: 0x0
      TCAM entry ID : 0x0 IPMC action: 0x4 FEC Accessed 1
      L3 Intf Refhandle : 0x308d76fee8 L3 interface ref key: 0x0
      Statistics enabled : not-set Statistics activated : not-set

      Egress Route OLEs:
        Handle: 308e27d930
```

```
Transaction ID: 103691
NPU ID: 0 Outgoing intf: Te0/0/0/1.1
OLE Type : Main Interface
outgoing port : 0x1d cud: 0x0 is_bundle: 0
Sys_port : 0x0 mpls encap id: 0x0 LAG ID: 0
is_pw_access: 0 pw_encap_id:0
L3 intf refhndl : 0x308cf40b08 L3 intf refkey: 0x4178
L2 Port refhandle : 0x308cf49358 L2 Port refkey: 0xe8
MPLS nh refhandle : 0x0 MPLS nh refkey: 0x0
LAG port refhandle : 0x0 LAG port refkey: 0x0
EFP-Visibility: not-set
Total fwd packets : 0 Total fwd bytes: 0

NPU ID: 0 Outgoing intf: Te0/0/0/0.1
OLE Type : Main Interface
outgoing port : 0x1e cud: 0x0 is_bundle: 0
Sys_port : 0x0 mpls encap id: 0x0 LAG ID: 0
is_pw_access: 0 pw_encap_id:0
L3 intf refhndl : 0x308cf3e668 L3 intf refkey: 0x4168
L2 Port refhandle : 0x308cf470a8 L2 Port refkey: 0xf0
MPLS nh refhandle : 0x0 MPLS nh refkey: 0x0
LAG port refhandle : 0x0 LAG port refkey: 0x0
EFP-Visibility: not-set
Total fwd packets : 0 Total fwd bytes: 0
```

Associated Commands

- interface tunnel-ip
- tunnel mode
- tunnel source
- tunnel destination

Use Case: Video Streaming

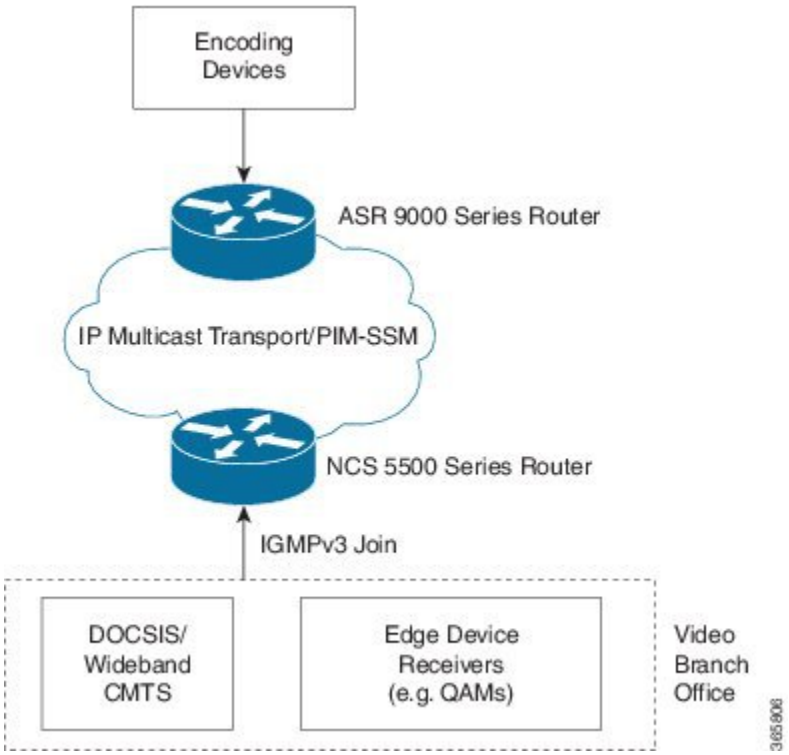
In today's broadcast video networks, proprietary transport systems are used to deliver entire channel line-ups to each video branch office. IP based transport network would be a cost efficient/convenient alternative to deliver video services combined with the delivery of other IP based services. (Internet delivery or business services)

By its very nature, broadcast video is a service well-suited to using IP multicast as a more efficient delivery mechanism to reach end customers.

The IP multicast delivery of broadcast video is explained as follows:

1. Encoding devices in digital primary headends, encode one or more video channels into a Moving Pictures Expert Group (MPEG) stream which is carried in the network via IP multicast.
2. Devices at video branch office are configured by the operator to request the desired multicast content via IGMP joins.
3. The network, using PIM-SSM as its multicast routing protocol, routes the multicast stream from the digital primary headend to edge device receivers located in the video branch office. These edge devices could be edge QAM devices which modulate the MPEG stream for an RF frequency, or CMTS for DOCSIS.

Figure 5: Video Streaming



Multicast Label Distribution Protocol

This section contains information related to Multicast Label Distribution Protocol (MLDP) and the associated features.

Multicast Label Distribution Protocol (MLDP) for Core

Table 9: Feature History Table

Feature Name	Release Information	Feature Description
MLDP Aggregated and Drop Statistics Measurement (v6 ingress stats on J2)	Release 7.3.1	This feature is now supported on routers that have the Cisco NC57 line cards installed and operate in compatible mode.

Multicast Label Distribution Protocol (MLDP) provides extensions to the Label Distribution Protocol (LDP) for the setup of point-to-multipoint (P2MP) and multipoint-to-multipoint (MP2MP) Label Switched Paths (LSPs) in Multiprotocol Label Switching (MPLS) networks.

MLDP eleminates the use of native multicast PIM to transport multicast packets across the core. In MLDP multicast traffic is label switched across the core. This saves a lot of control plane processing effort.

Characteristics of MLDP Profiles on Core

The following MLDP profiles are supported when the router is configured as a core router:

- Profile 5—Partitioned MDT - MLDP P2MP - BGP-AD - PIM C-mcast Signaling
- Profile 6—VRF MLDP - In-band Signaling
- Profile 7—Global MLDP In-band Signaling
- Profile 12—Default MDT - MLDP - P2MP - BGP-AD - BGP C-mcast Signaling
- Profile 14—Partitioned MDT - MLDP P2MP - BGP-AD - BGP C-mcast Signaling
- Profile 17—Default MDT - MLDP - P2MP - BGP-AD - PIM C-mcast Signaling
- Profile 22—RSVP-TE P2MP

Point-to-Multipoint Profiles on Core and Edge Routers

The following profiles are supported when the router is configured as a core router and edge router for p2mp:

- Profile 8—Global P2MP-TE
- Profile 10—VRF Static-P2MP-TE with BGP AD

Multicast MLDP for Edge Router

The following MLDP and P2MP-TE profiles are supported when the router is configured as an edge router:

- Profile 6—VRF MLDP - In-Band Signaling
- Profile 7—Global MLDP In-band Signaling
- Profile 8—Global Static - P2MP-TE
- Profile 10—VRF Static - P2MP TE - BGP-AD
- Profile 14—MLDP Partitioned MDT P2MP with BGP AD and BGP-C Multicast Signaling
- Profile 22—RSVP-TE P2MP

Multicast MLDP Profile 14 support on an Edge Router

Table 10: Feature History Table

Feature Name	Release Information	Feature Description
PIM SM for mVPN Profile 14	Release 7.5.1	With this feature, MVPN profile 14 is now extended to support PIM SM mode for IPv4 and static RP. PIM SM for MVPN is not supported on Cisco NC57 line cards.

Feature Name	Release Information	Feature Description
MLDP Profile 14 support on an Edge Router	Release 7.3.1	This feature is now supported on Cisco NCS 5500 routers and the NCS57 line cards installed and operate in native and compatible mode.

The MLDP Profile 14 is supported when the router is configured as an edge router.

IP based transport network is a cost efficient and convenient alternative to deliver video services combined with the delivery of other IP based services. To deliver IPTV content MLDP Profile 14 also called as the partitioned MDT, is supported when a router is configured as an edge router.

These are the characteristics of the profile 14:

- Customer traffic is SSM.
- PIM SM mode for IPv4 and static RP.

Note PIM SM for MVPN is not supported on Cisco NC57 line cards.

- Inter-AS Option A, B and C is supported.
- All PEs must have a unique BGP Route Distinguisher (RD) value.

Configuration Example for mLDP Profile 14 on Edge Routers

```
vrf one
 address-family ipv4 unicast
  import route-target
   1:1
  !
  export route-target
   1:1
  !
 !

router pim
 vrf one
  address-family ipv4
   rpf topology route-policy rpf-for-one
   mdt c-multicast-routing bgp
   !
   interface GigabitEthernet0/1/0/0
    enable
   !
  !
 !
!

route-policy rpf-for-one
  set core-tree mldp-partitioned-p2mp
end-policy
!
```

```
multicast-routing
 vrf one
  address-family ipv4
   mdt source Loopback0
   mdt partitioned mldp ipv4 p2mp
   rate-per-route
   interface all enable
   bgp auto-discovery mldp
   !
   accounting per-prefix
  !
 !
!

mpls ldp
 mldp
  logging notifications
  address-family ipv4
  !
 !
!
```

Label Switched Multicast (LSM) Multicast Label Distribution Protocol (mLDP) based Multicast VPN (mVPN) Support

Table 11: Feature History Table

Feature Name	Release Information	Feature Description
MVPN Support	Release 7.3.1	This feature is now supported on routers that have the Cisco NC57 line cards installed and operate in compatible mode.

Label Switch Multicast (LSM) is MPLS technology extensions to support multicast using label encapsulation. Next-generation MVPN is based on Multicast Label Distribution Protocol (mLDP), which can be used to build P2MP and MP2MP LSPs through a MPLS network. These LSPs can be used for transporting both IPv4 and IPv6 multicast packets, either in the global table or VPN context. mLDP is supported on both core and edge routers.

When router is positioned as the core router running mLDP, it only supports the Profiles 5, 6, 7, 12, 14, and 17 irrespective of the profiles supported on the edge router.

When router is positioned as the edge router running mLDP, it only supports the Profiles 6, 7, and 14.

For more information about the characteristics of each of the mLDP Profiles, Characteristics of MLDP Profiles on Core

Benefits of LSM MLDP based MVPN

LSM provides these benefits when compared to GRE core tunnels that are currently used to transport customer traffic in the core:

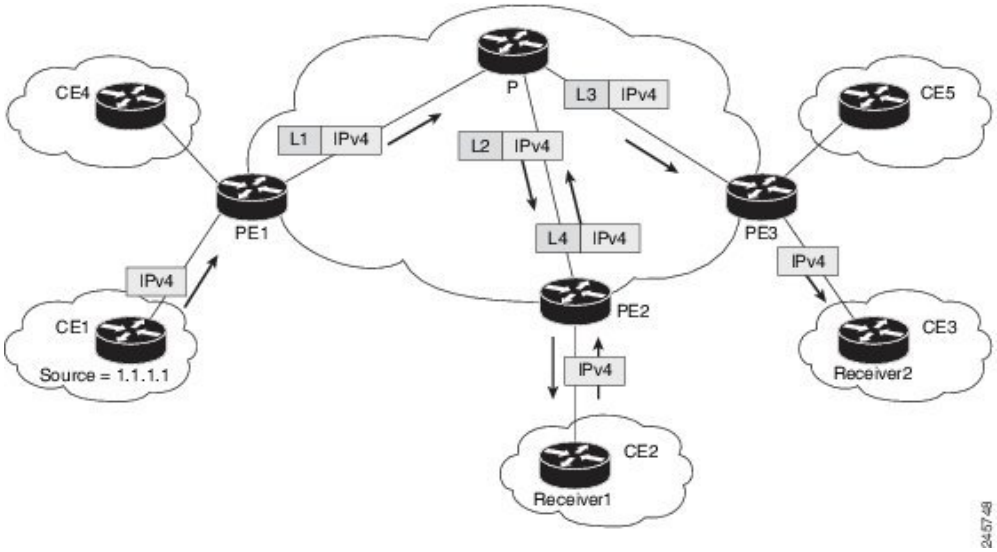
- It leverages the MPLS infrastructure for transporting IP multicast packets, providing a common data plane for unicast and multicast.
- It applies the benefits of MPLS to IP multicast such as Fast ReRoute (FRR) and

- It eliminates the complexity associated PIM.

Configuring MLDP MVPN

The MLDP MVPN configuration enables IPv4 multicast packet delivery using MPLS. This configuration uses MPLS labels to construct default and data Multicast Distribution Trees (MDTs). The MPLS replication is used as a forwarding mechanism in the core and edge network. For MLDP MVPN configuration to work, ensure that the global MPLS MLDP configuration is enabled. To configure MVPN extranet support, configure the source multicast VPN Routing and Forwarding (mVRF) on the receiver Provider Edge (PE) router or configure the receiver mVRF on the source PE. MLDP MVPN is supported for both intranet and extranet.

Figure 6: MLDP based MPLS Network for Core and Edge Routers



Packet Flow in mLDP-based Multicast VPN

For each packet coming in, MPLS creates multiple out-labels. Packets from the source network are replicated along the path to the receiver network. The CE1 router sends out the native IP multicast traffic. The Provider Edge1 (PE1) router imposes a label on the incoming multicast packet and replicates the labeled packet towards the MPLS core network. When the packet reaches the core router (P), the packet is replicated with the appropriate labels for the MP2MP default MDT or the P2MP data MDT and transported to all the egress PEs. Once the packet reaches the egress PE (edge routers), the label is removed and the IP multicast packet is replicated onto the VRF interface. Basically, the packets are encapsulated at headend and decapsulated at tailend on the PE routers.

Realizing a mLDP-based Multicast VPN

There are different ways a Label Switched Path (LSP) built by mLDP can be used depending on the requirement and nature of application such as:

- P2MP LSPs for global table transit Multicast using in-band signaling.
- P2MP/MP2MP LSPs for MVPN based on MI-PMSI or Multidirectional Inclusive Provider Multicast Service Instance (Rosen Draft).

- P2MP/MP2MP LSPs for MVPN based on MS-PMSI or Multidirectional Selective Provider Multicast Service Instance (Partitioned E-LAN).

The router performs the following important functions for the implementation of MLDP:

1. Encapsulating VRF multicast IP packet with GRE/Label and replicating to core interfaces (imposition node).
2. Replicating multicast label packets to different interfaces with different labels (Mid node).
3. Decapsulate and replicate label packets into VRF interfaces (Disposition node).

Characteristics of mLDP Profiles

The characteristics of various mLDP profiles are listed in this section.

Configuration rules for profiles

MLDP inband signaling

MLDP Inband signaling allows the core to create (S,G) or (*,G) state without using out-of-band signaling such as BGP or PIM. It is supported in VRF (and in the global context). Both IPv4 and IPv6 multicast groups are supported.

In MLDP Inband signaling, one can configure an ACL range of multicast (S,G). This (S,G) can be transported in MLDP LSP. Each multicast channel (S,G), is 1 to 1 mapped to each tree in the inband tree. The (S,G) join, through IGMP/MLD/PIM, will be registered in MRIB, which is the client of MLDP.

MLDP In-band signalling supports transiting PIM (S,G) or (*,G) trees across a MPLS core without the need for an out-of-band protocol. In-band signaling is only supported for shared-tree-only forwarding (also known as sparse-mode threshold infinity). PIM Sparse-mode behavior is not supported (switching from (*,G) to (S,G).

The details of the MLDP profiles are discussed in the *Multicast Configuration Guide for Cisco NCS 5500 Series Routers*

Restrictions for mLDP on Edge Routers

The restrictions applicable for mLDP on edge routers are as follows:

- NETCONF/YANG on MVPN for Profile 6 and Profile 7 is not supported.
- MLDP ping traceroute is not supported.
- IPv6 BVI is not supported.
- Netflow for MPLS-encapsulated multicast packets is not supported.
- MLDP Fast-Reroute is supported for Profile 14 only.

Configuration Process for MLDP MVPN (Intranet)

These steps provide a broad outline of the different configuration process of MLDP MVPN for intranet:

- Enabling MPLS MLDP
    - configure

    - mpls ldp mldp

- Configuring a VRF entry
    - configure
    - vrf vrf_name
    - address-family ipv4/ipv6 unicast
    - import route-target route-target-ext-community
    - export route-target route-target-ext-community
- Configuring VPN ID
    - configure
    - vrf *vrf_name*
    - vpn id *vpn_id*
- Configuring MVPN Routing and Forwarding instance
    - configure
    - multicast-routing vrf *vrf_name*
    - address-family ipv4
    - mdt default mldp ipv4 *root-node*
- Configuring the Route Distinguisher
    - configure
    - router bgp *AS Number*
    - vrf *vrf_name*
    - rd *rd_value*
- Configuring Data MDTs (optional)
    - configure
    - multicast-routing vrf *vrf_name*
    - address-family ipv4
    - mdt data <1-255>
- Configuring BGP MDT address family
    - configure
    - router bgp *AS Number*
    - address-family ipv4 mdt

- Configuring BGP vpnv4 address family
  - configure
  - router bgp *AS Number*
  - address-family vpnv4 unicast
- Configuring BGP IPv4 VRF address family
  - configure
  - router bgp *AS Number*
  - vrf *vrf_name*
  - address-family ipv4 unicast
- Configuring PIM SM/SSM Mode for the VRFs
  - configure
  - router pim
  - vrf *vrf_name*
  - address-family ipv4
  - rpf topology route-policy *rosen_mvpn_mldp*

  For each profile, a different route-policy is configured.
- Configuring route-policy
  - route-policy *rosen_mvpn_mldp*
  - set core-tree *tree-type*
  - pass
  - end-policy

Note The configuration of the above procedures depends on the profile used for each configuration.

Configuration Example for MLDP on Core

```
mpls ldp
 mldp
  logging notifications
  address-family ipv4
  !
 !
!
```

Flexible Algorithm for MLDP

Table 12: Feature History Table

Feature Name	Release Information	Feature Description
Flexible Algorithm for Multicast VPN profiles	Release 7.5.2	Flexible Algorithm is now available for the following profiles: • Profile 12: Default MDT - MLDP - P2MP - BGP-AD - BGP C-Mcast Signaling • Profile 14: Partitioned MDT - MLDP P2MP - BGP-AD - BGP C-Mcast Signaling
Flexible Algorithm for MLDP	Release 7.5.1	This feature gives you the flexibility to customize the metrics that IGP uses to route traffic for MLDP tunnels. With this feature, your router can generate two multicast streams for the same feed, thus ensuring low latency and high availability of multicast traffic. This feature introduces the **flex-algoflex-algo** keyword.

IGP determines the shortest path to send traffic through MLDP tunnels. However, at times, you may require to choose a path other than the shortest one. For instance, when you want to achieve low latency or want to send the traffic on a specific path to avoid a set of links or build totally two or more disjoint paths.

Multipoint LDP (mLDP) flexible algorithm allows you to customize the IGP path computation based on the business needs.

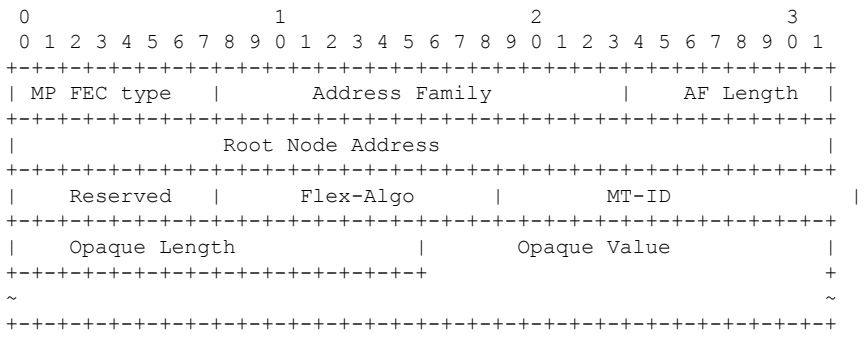
With this feature, you can get disjoint paths for two multicast streams. Each stream carries traffic within a separate network and helps reduce a connection loss or delay ensuring low latency and high availability of multicast traffic. This feature allows you to segregate multicast traffic to specific regions.

mLDP flexible algorithm is based on Segment Routing flexible algorithm that allows operators to customize IGP shortest path computation according to their own needs. For more information, see Enabling Segment Routing Flexible Algorithm.

To compute two different paths, the mLDP flexible algorithm uses a 2-tuple hash algorithm, which includes MPLS Multi-Topology Identifier (MT-ID) and IGP algorithm.

MPLS multi topology Identifier (MT-ID) is a unique identifier that is used to associate an LSP with multi topology. This identifier is part of the mLDP FEC encoding so that LDP peers are able to set up an MP LSP through their own defined policy and avoid conflicting policies for the same mLDP FEC. The MT-ID and IGP Algorithm must be part of the FEC so that different values result in unique MP-LSP FEC elements.

mLDP flexible algorithm is stored in IGP Algorithm (IPA) Registry field. A 16-bit reserved field is included and out of which 8 bits are used for flexible algorithm. The root is an IP address identifying the Root of an MLDP-P2MP tree. Only IPv4 address is supported.



For example, two flexible algorithms are required to implement a disjoint-paths scenario.

Leafs belonging to the first plane are associated with flexible algorithm 130 and leafs belonging another plane are associated with flexible algorithm 128.

mLDP with flexible algorithm also provides the following support:

- Profile 12: Default MDT - MLDP - P2MP - BGP-AD - BGP C-Mcast Signaling
- Profile 14: Partitioned MDT - MLDP P2MP - BGP-AD - BGP C-Mcast Signaling
- MLDP Loop-Free Alternative Fast Reroute
- Data Multicast Distribution Tree (MDT) route policy supports `none` keyword.

Restrictions

The following features are not supported with flexible algorithm:

- PIM BiDir in the core
- Carrier supporting carrier (CSC)
- Inter-AS
- Extranet
- Default MDT
- MVPN profiles except profile-14

How to configure mLDP Flexible Algorithm?

Configure partitioned MDT with flexible algorithm MVPN profile:

```
Router #configure
Router(config)# multicast-routing
Router(config-mcast)# vrf red
Router(config-mcast-red)# address-family ipv4
Router(config-mcast-red-ipv4)# mdt partitioned mldp ipv4 p2mp flex-algo 128
```

Configure data MDT with flexible algorithm:

```
Router# configure
Router(config)# multicast-routing
```

```
Router(config-mcast)# vrf red
Router(config-mcast-red)# address-family ipv4
Router(config-mcast-red-ipv4)# mdt data mldp 10 flex-algo 130
```

Configure route policy for Data MDT with flexible algorithm:

```
Router # configure
Router(config)# multicast-routing vrf red address-family ipv4
Router(config-mcast-red-ipv4)# mdt data mldp 10 route-policy rpl-for-red flex-algo 130
```

Note We recommend having 1:1 allocation of Data MDTs with number of flows. However, you can configure less Data MDTs than the number of flows. But you may not see the expected results. You cannot modify the flexible algorithm configuration without removing the MDT configuration. You must reconfigure MDT with the new flexible algorithm. Other combinations like no flexible algorithm to flexible algorithm are not supported.

Configuration Example

The route-policy and flexible algorithm are configured on Data MDT CLI.

In this example,

- Multicast groups 232.1.1.1 and 232.1.1.2 match the route policy it uses the Data MDT created with flexible algorithm 128.
- Multicast groups 232.1.1.3 and 232.1.1.4 also match the route-policy it uses the Data MDT created with flexible algorithm 129.

  All other groups do not match the route-policy so it uses the Data MDT created with flexible algorithm 130 which is configured in Data MDT CLI.

```
route-policy c1-data-mdt
  if destination in (232.1.1.1, 232.1.1.2) then
    set flex-algo 128
  elseif destination in (232.1.1.3, 232.1.1.4) then
    set flex-algo 129
  endif
end-policy
!
multicast-routing
 vrf red
  address-family ipv4
   mdt partitioned mldp ipv4 p2mp flex-algo 131
   mdt data mldp 100 route-policy c1-data-mdt flex-algo 130
  !
 !
!
```

In the following example, route-policy is configured on Data MDT CLI but without flexible algorithm.

```
route-policy c1-data-mdt
  if destination in (232.1.1.1, 232.1.1.2) then
    set flex-algo 128
  elseif destination in (232.1.1.3, 232.1.1.4) then
    set flex-algo 129
```

```
  endif
end-policy
!
multicast-routing
 vrf red
  address-family ipv4
   mdt partitioned mldp ipv4 p2mp flex-algo 131
   mdt data mldp 100 route-policy c1-data-mdt
  !
 !
!
```

Flexible algorithm is configured on Data MDT CLI and all groups uses the Data MDT created with flexible algorithm 130.

```
multicast-routing
 vrf red
  address-family ipv4
   mdt partitioned mldp ipv4 p2mp flex-algo 131
   mdt data mldp 100 flex-algo 130
```

In the following example, flexible algorithms are configured in Partitioned MDT. All groups use the Data MDT created with Flexible Algorithm 131 which is configured in Partitioned MDT CLI.

```
multicast-routing
 vrf red
  address-family ipv4
   mdt partitioned mldp ipv4 p2mp flex-algo 131
   mdt data mldp 100
```

Verification

```
Router# show mvpn vrf red context private

MVPN context information for VRF red (0xa99dbf8)

RD: 1:1 (Valid, IID 0x1), VPN-ID: 0:0
Import Route-targets : 2
  RT:10.0.0.4:0, BGP-AD
  RT:10.0.0.4:17, BGP-AD
BGP Auto-Discovery Enabled (I-PMSI added) , MS-PMSI sent
MLDP Core-tree data:
  MDT Name: Lmdtred, Handle: 0x8041b0, idb: 0xa9b1c18
  MTU: 1376, MaxAggr: 255, SW_Int: 30, AN_Int: 60
  RPF-ID: 9/0, C:0, O:1, D:0, CP:0
  MLDP Number of Roots: 0 (Local: 0), HLI: 0x00000, Rem HLI: 0x00000
  Discovery Type,Value: Flex-Algo,131
  Data MDT Discovery Type,Value: Flex-Algo,130
  Partitioned MDT: Configured, P2MP (RD:Not added, ID:Added), HLI: 0x00005, Loc Label:
24017, Remote: None
    ID: 2 (0xa79ce90), Ctrl Trees : 0/0/0, Ctrl ID: 0 (0x0), IR Ctrl ID: 0 (0x0), Ctrl HLI:
 0x00000
  P2MP Def MDT ID: 0 (0x0), added: 0, HLI: 0x00000, Cfg: 0/0


Router# show mpls mldp database root 10.0.0.21 opaquetype global-id 5

mLDP database
LSM-ID: 0x00020  Type: P2MP  Uptime: 2d17h
  FEC Root           : 10.0.0.21
  IGP Algorithm      : Flex-Algo 129
   Opaque decoded     : [global-id 5]
```

```
  Upstream neighbor(s) :
      10.0.0.4:0 [Active] Uptime: 1d22h
      Local Label (D) : 24040
  Downstream  client(s):
    PIM MDT            Uptime: 2d17h
      Egress intf     : LmdtWHITE
      Table ID        : IPv4: 0xe0000012 IPv6: 0xe0800012
      RPF ID          : 15
      RD              : 257:18154475
```

```
Router# show pim vrf vpn1 mdt cache
Core Source      Cust (Source, Group)              Core Data        Expires
5.15.15.15       (100.0.1.1, 233.1.1.1)            [global-id 17, Flex-Algo 129]  never

5.15.15.15       (100.0.1.1, 233.1.1.2)            [global-id 18, Flex-Algo 129]  never

5.15.15.15       (100.0.1.1, 233.1.1.3)            [global-id 19, Flex-Algo 129]  never

5.15.15.15       (100.0.1.1, 233.1.1.4)            [global-id 20, Flex-Algo 129]  never

5.15.15.15       (100.0.1.1, 233.1.1.5)            [global-id 21, Flex-Algo 129]  never
```

Configure VRF MLDP In-Band Signaling on Edge Routers

To configure VRF MLDP in-band signaling (Profile 6) on edge routers, you must complete the following tasks:

1. Assign a route policy in PIM to select a reverse-path forwarding (RPF) topology.
2. Configure route policy to set the Multicast Distribution Tree (MDT) type to MLDP inband.
3. Enable MLDP-inband signaling in multicast routing.
4. Enable MPLS for MLDP.

Configuration

/* Assign a route policy in PIM to select a reverse-path forwarding (RPF) topology */

```
RP/0/RP0/CPU0:router(config)#router pim
RP/0/RP0/CPU0:router(config-pim)#vrf one
RP/0/RP0/CPU0:router(config-pim-one)#address-family ipv4
RP/0/RP0/CPU0:router(config-pim-one-ipv4)#rpf topology route-policy rpf-vrf-one
```

/* Configure route policy to set the MDT type to MLDP inband */

```
RP/0/RP0/CPU0:router(config)#route-policy rpf-vrf-one
RP/0/RP0/CPU0:router(config-rpl)#set core-tree mldp-inband
RP/0/RP0/CPU0:router(config-rpl)#end-policy
```

/* Enable MLDP-inband signaling in multicast routing */

```
RP/0/RP0/CPU0:router(config)#multicast-routing
RP/0/RP0/CPU0:router(config-mcast)#vrf one
RP/0/RP0/CPU0:router(config-mcast-one)#address-family ipv4
RP/0/RP0/CPU0:router(config-mcast-one-ipv4)#mdt source loopback 0
```

```
RP/0/RP0/CPU0:router(config-mcast-one-ipv4)#mdt mldp in-band-signaling ipv4
RP/0/RP0/CPU0:router(config-mcast-one-ipv4)#interface all enable
```

/* Enable MPLS MLDP */

```
RP/0/RP0/CPU0:router(config)#mpls ldp
RP/0/RP0/CPU0:router(config-ldp)#mldp
```

Configure Global MLDP In-band Signaling on Edge Routers

To configure global MLDP in-band signaling (Profile 7) on edge routers, you must complete the following tasks:

1. Assign a route policy in PIM to select a reverse-path forwarding (RPF) topology.
2. Configure route policy to set the MDT type to MLDP Inband.
3. Enable MLDP inband signaling in multicast routing.
4. Enable MPLS MLDP.

Configuration

/* Assign a route policy in PIM to select a reverse-path forwarding (RPF) topology */

```
RP/0/RP0/CPU0:router(config)#router pim
RP/0/RP0/CPU0:router(config-pim)#address-family ipv4
RP/0/RP0/CPU0:router(config-pim-default-ipv4)#rpf topology route-policy rpf-global
RP/0/RP0/CPU0:router(config-pim-default-ipv4)#interface TenGigE 0/0/0/21
RP/0/RP0/CPU0:router(config-pim-ipv4-if)#enable
```

/* Configure route policy to set the MDT type to MLDP inband */

```
RP/0/RP0/CPU0:router(config)#route-policy rpf-global
RP/0/RP0/CPU0:router(config-rpl)#set core-tree mldp-inband
RP/0/RP0/CPU0:router(config-rpl)#end-policy
```

/* Enable MLDP-inband signaling in multicast routing */

```
RP/0/RP0/CPU0:router(config)#multicast-routing
RP/0/RP0/CPU0:router(config-mcast)#address-family ipv4
RP/0/RP0/CPU0:router(config-mcast-default-ipv4)#interface loopback 0
RP/0/RP0/CPU0:router(config-mcast-default-ipv4-if)#enable
RP/0/RP0/CPU0:router(config-mcast-default-ipv4-if)#exit
RP/0/RP0/CPU0:router(config-mcast-default-ipv4)#mdt source loopback 0
RP/0/RP0/CPU0:router(config-mcast-default-ipv4)#mdt mldp in-band-signaling ipv4
RP/0/RP0/CPU0:router(config-mcast-default-ipv4)#interface all enable
```

/* Enable MPLS MLDP */

```
RP/0/RP0/CPU0:router(config)#mpls ldp
RP/0/RP0/CPU0:router(config-ldp)#mldp
```

Configuration Examples for Inband mLDP Profiles on Edge Routers

Running Configuration for VRF MLDP In-Band Signaling (Profile 6)

```
router pim
 vrf one
  address-family ipv4
```

```
   rpf topology route-policy rpf-vrf-one

   route-policy rpf-vrf-one
     set core-tree mldp-inband
   end-policy

multicast-routing
 vrf one
  address-family ipv4
   mdt source Loopback0
   mdt mldp in-band-signaling ipv4
   interface all enable

mpls ldp
 mldp
```

Running Configuration for Global MLDP In-band Signaling (Profile 7)

```
router pim
  address-family ipv4
   rpf topology route-policy rpf-global
   interface TenGigE0/0/0/21
    enable

route-policy rpf-global
  set core-tree mldp-inband
end-policy

multicast-routing
 address-family ipv4
  interface Loopback0
   enable
  !
  mdt source Loopback0
  mdt mldp in-band-signaling ipv4
  interface all enable
 !
mpls ldp
 mldp
```

Verification of MLDP Configuration on Edge Routers

Use the following commands to verify the MLDP configuration on edge routers.

To check the MLDP neighbors, use the **show mpls mldp neighbor** command.

```
RP/0/RP0/CPU0:Head# show mpls mldp neighbors
mLDP neighbor database
MLDP peer ID      : 2.2.2.2:0, uptime 07:47:59 Up,
  Capabilities     : GR, Typed Wildcard FEC, P2MP, MP2MP
  Target Adj       : No
  Upstream count   : 1
  Branch count     : 1
  LDP GR           : Enabled
                   : Instance: 1
  Label map timer  : never
  Policy filter in :
  Path count       : 1
  Path(s)          : 12.1.1.2          TenGigE0/0/1/0/3.2000 LDP
  Adj list         : 12.1.1.2          TenGigE0/0/1/0/3.2000
  Peer addr list   : 2.25.32.2
```

```
                    : 2.2.2.2
                    : 11.1.1.1
                    : 12.1.1.2
                    : 13.10.1.1
```

To display the contents of the Label Information Base (LIB), use the **show mpls mldp bindings** command.

```
RP/0/RP0/CPU0:Head#show mpls mldp bindings
mLDP MPLS Bindings database

LSP-ID: 0x00001 Paths: 7 Flags:
0x00001 P2MP   5.5.5.5 [vpnv6 1:1 2015:1:1::3 ff3e::1]
   Local Label: 70009
   Remote Label: 64018 NH: 12.1.1.2 Inft: TenGigE0/0/1/0/3.2000
   Remote Label: 64022 NH: 50.1.1.1 Inft: TenGigE0/0/1/3/0
   Remote Label: 30002 NH: 30.10.1.2 Inft: Bundle-Ether56
   Remote Label: 64023 NH: 60.1.1.2 Inft: HundredGigE0/0/1/1
   Remote Label: 64024 NH: 70.1.1.1 Inft: TenGigE0/0/1/2/0
   Remote Label: 64022 NH: 40.1.1.1 Inft: TenGigE0/0/0/18
```

To display the MLDP event traces, use the **show mpls mldp trace** command.

```
RP/0/RP0/CPU0:Head#show mpls mldp trace
3535 wrapping entries (631040 possible, 35584 allocated, 0 filtered, 3535 total)
May 30 23:30:21.121 MLDP GLO 0/RP0/CPU0 t6746 GEN  : Trace pre-init iox success
May 30 23:30:21.121 MLDP GLO 0/RP0/CPU0 t6746 GEN  : Debug pre-init iox success
May 30 23:30:21.121 MLDP GLO 0/RP0/CPU0 t6746 GEN  : API pre-init iox success
May 30 23:30:21.121 MLDP GLO 0/RP0/CPU0 t6746 GEN  : Bitfield pre-init iox success
May 31 12:08:39.465 MLDP GLO 0/RP0/CPU0 t6746 GEN  : mldp_evm 0x563de8f01698 allocated
May 31 12:08:39.465 MLDP GLO 0/RP0/CPU0 t6746 GEN  : EVM init iox success
May 31 12:08:39.472 MLDP GLO 0/RP0/CPU0 t6746 GEN  : Registered EDM on active success
May 31 12:08:39.472 MLDP GLO 0/RP0/CPU0 t6746 GEN  : EDM Ac/St init iox again
May 31 12:08:39.472 MLDP GLO 0/RP0/CPU0 t6746 GEN  : Registered EDM Location on active
success
May 31 12:08:39.472 MLDP GLO 0/RP0/CPU0 t6746 GEN  : EDM Loc init iox success
May 31 12:08:39.475 MLDP GLO 0/RP0/CPU0 t6746 GEN  : LMRIB init iox success
May 31 12:08:39.475 MLDP GLO 0/RP0/CPU0 t18944 MRIB : MRIB connection established
May 31 12:08:39.475 MLDP GLO 0/RP0/CPU0 t6746 GEN  : Interface manager init iox success
May 31 12:08:39.475 MLDP GLO 0/RP0/CPU0 t6746 GEN  : Async init iox success
May 31 12:08:39.475 MLDP GLO 0/RP0/CPU0 t6746 GEN  : Boolean init iox success
May 31 12:08:39.475 MLDP GLO 0/RP0/CPU0 t6746 GEN  : Timers init iox success
May 31 12:08:39.479 MLDP GLO 0/RP0/CPU0 t6746 GEN  : RUMP init iox success
May 31 12:08:39.479 MLDP GLO 0/RP0/CPU0 t6746 GEN  : Chunks init iox success
May 31 12:08:39.509 MLDP ERR 0/RP0/CPU0 t6746 RIB  : RIB not ready
May 31 12:08:39.509 MLDP ERR 0/RP0/CPU0 t6746 RIB  : RIB not ready
May 31 12:08:39.512 MLDP GLO 0/RP0/CPU0 t6746 GEN  : mldp_ens_event_ctx_chunk is NULL
May 31 12:08:39.512 MLDP GLO 0/RP0/CPU0 t6746 GEN  : Context Table init iox success
May 31 12:08:39.512 MLDP GLO 0/RP0/CPU0 t6746 GEN  : mldp_rib_main_evm 0x563de8fd23e8
allocated
May 31 12:08:39.512 MLDP GLO 0/RP0/CPU0 t6746 GEN  : RIB Thread EVM init rib success
May 31 12:08:39.512 MLDP GLO 0/RP0/CPU0 t6746 GEN  : RIB Thread Chunk init rib success
May 31 12:08:39.512 MLDP GLO 0/RP0/CPU0 t6746 GEN  : RIB Thread queue init rib success
May 31 12:08:39.512 MLDP GLO 0/RP0/CPU0 t6746 RIB  : Bound to RIB, fd: 354
```

MLDP Loop-Free Alternative Fast Reroute

Table 13: Feature History Table

Feature Name	Release Information	Feature Description
Flexible Algorithm for MLDP Loop-Free Alternative Fast Reroute	Release 7.5.2	You can build disjoint live-live paths or create specific paths with flexible algorithm constraints, have low-latency routing without IGP constraints. MLDP route-policy supporting flexible algorithm-based filtering, which provides more granular enablement of FRR for LSPs, is also available.
MLDP Loop-Free Alternative Fast Reroute	Release 7.5.1	This feature is now supported on routers that have the Cisco NC57 line cards installed and operate in native and compatible mode. With this feature, the router can quickly switch traffic to a precomputed loop-free alternative (LFA) path by allocating a label to the incoming traffic. This minimizes the traffic loss ensuring fast convergence.
MLDP Loop-Free Alternative Fast Reroute	Release 7.3.1	In the event of a link failure, this feature enables the router to quickly switch traffic to a precomputed loop-free alternative (LFA) path by allocating a label to the incoming traffic. This minimizes the traffic loss ensuring fast convergence.

Generally, in a network, a network topology change, caused by a failure in a network, results in a loss of connectivity until the control plane convergence is complete. There can be various levels of loss of connectivity depending on the performance of the control plane, fast convergence tuning, and leveraged technologies of the control plane on each node in the network.

The amount of loss of connectivity impacts some loss-sensitive applications, which have severe fault tolerance (typically of the order of hundreds of milliseconds and up to a few seconds). To ensure that the loss of connectivity conforms to such applications, a technology implementation for data plane convergence is essential. **Fast Reroute (FRR)** is one of such technologies that is primarily applicable to the network core.

With the FRR solution, at each node, the backup path is precomputed, and the traffic is routed through this backup path. As a result, the reaction to failure is local; immediate propagation of the failure and subsequent processing on to other nodes is not required. With FRR, if the failure is detected quickly, a loss of connectivity as low as 10s of milliseconds is achieved.

Loop-Free Alternative Fast Reroute

IP Loop Free Alternative FRR is a mechanism that enables a router to rapidly switch traffic to a pre-computed or a pre-programmed **loop-free alternative (LFA)** path (Data Plane Convergence), following either an adjacent link and node failure, or an adjacent link or node failure in both IP and LDP networks. The LFA path is used to switch traffic till the router installs the new primary next-hops based on the changed network topology (Control Plane Convergence).

The goal of LFA FRR is to reduce the loss of connectivity to tens of milliseconds by using a pre-computed alternative next-hop, in the case where the selected primary next-hop fails.

There are two approaches to computing LFA paths:

- **Link-based (per-link):** In link-based LFA paths, all prefixes reachable through the primary (protected) link share the same backup information. This means that the whole set of prefixes sharing the same primary also shares the repair and FRR ability.
- **Prefix-based (per-prefix):** Prefix-based LFAs allow computing backup information for each prefix. This means that the repair and backup information computed for a given prefix using prefix-based LFA may be different from the one computed by link-based LFA.

  Node-protection support is available with per-prefix LFA FRR on ISIS currently. It uses a tie-breaker mechanism in the code to select node-protecting backup paths.

The per-prefix LFA approach is preferred to the per-link LFA approach for the following reasons:

- Better node failure resistance.
- Better coverage: Each prefix is analyzed independently.
- Better capacity planning: Each flow is backed up on its own optimized shortest path.

MLDP LFA FRR

The point-to-point physical or bundle interface FRR mechanism is supported on MLDP. FRR with LFA backup is also supported on MLDP. When there is a link failure, MLDP automatically sets up and chooses the backup path. With this implementation, you must configure the physical or bundle interface for unicast traffic, so that the MLDP can act as an MLDP FRR.

LFA FRR support on MLDP is a per-prefix backup mechanism. As part of computing the LFA backup for a remote IP, the LFA backup paths for the loopback address of the downstream intermediate nodes are also computed. MLDP uses this small subset of information, by using the loopback address of the peer to compute the LFA backup path.

Note Both IPv4 and IPv6 traffic is supported on the MLDP LFA FRR solution.

For information on use cases, see the MLDP with Flex-Algo in Service Provider Networks White Paper.

MLDP LFA FRR with Flexible Algorithm

The MLDP LFA FRR with Flexible Algorithm uses the segment routed (SR) LFA FRR-selected primary and backup paths to the peers and emulates a multicast distribution tree, instead of multicast label-switched paths (LSP). It helps in having a more efficient FRR with low-latency routing, live-live disjoint paths, or constraining

multicast flows to a specific region. Interior Gateway Protocol (IGP) calculates LFA path for each learned node SID within the IGP domain.

Note All the following limitations of MLDP LFR FRR without Flexible Algorithm also apply to MLDP LFA FRR with Flexible Algorithm:

- Node protection is not supported.

Supported MLDP Profiles with Flexible Algorithm

The list of supported MLDP profiles is the following

- Profile 12: Default MDT - MLDP - P2MP - BGP-AD - BGP C-Mcast Signaling
- Profile 14: Partitioned MDT - MLDP P2MP - BGP-AD - BGP C-Mcast Signaling

Supported MLDP Profiles

The following MLDP profile is supported:

- Profile 14: Partitioned MDT - MLDP P2MP - BGP-AD - BGP C-Mcast Signaling

Advantages of LFA FRR

The following are the advantages of the LFA FRR solution:

- The backup path for the traffic flow is pre-computed.
- Reaction to failure is local, an immediate propagation and processing of failure on to other nodes is not required.
- If the failure is detected in time, the loss of connectivity of up to 10s of milliseconds can be achieved. Prefix independency is the key for a fast switchover in the forwarding table.
- The mechanism is locally significant and does not impact the Interior Gateway Protocol (IGP) communication channel.
- LFA next-hop can protect against:
  - a single link failure
  - failure of one of more links within a shared risk link group (SRLG)
  - any combination of the above

MLDP LFA FRR - Features

The following are the features of mLDP LFA FRR solution:

- Supports both IPv4 and IPv6 traffic
- Supports Profile 14 mLDP profile
- Supports the LAG interfaces and sub-interfaces in the core

- Supports both ISIS and OSPF routing protocols
- Supports switchover time of less than 50 milliseconds
- Supports switchover time to be independent of the number of multicast routes that has to be switched over

Note ECMP primary and backup paths are not supported.

Limitations of LFA FRR

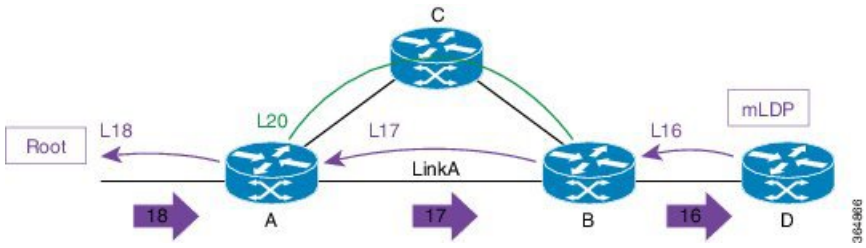
The following are some of the known limitations of the LFA FRR solution:

- When a failure that is more extensive than that which the alternate was intended to protect occurs, there is the possibility of temporarily looping traffic (micro looping until Control Plane Convergence).
- Topology dependent. For example, either MPLS or MLDP dependent.
- Complex implementation.
- The solution is currently not supported on all platforms.
- MLDP FRR over TI-LFA is currently not supported.

MLDP LFA FRR - Working

To enable FRR for mLDP over physical or bundle interfaces, LDP session-protection has to be configured. The sequence of events that occur in an mLDP LFA FRR scenario is explained with the following example:

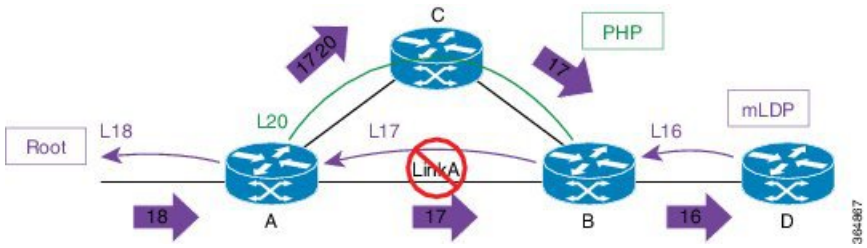
Figure 7: MLDP LFA FRR - Setup



In this figure:

1. Router A is the source provider edge router, and the next Hop is Router B.
2. The primary path is Router A -> Router B - > Router D, and the backup path is from Router A -> Router C -> Router B -> Router D. The backup path is pre-computed by IGP through LFA prefix-based selection.
3. MLDP LSP is build from D, B, and A towards the root.
4.

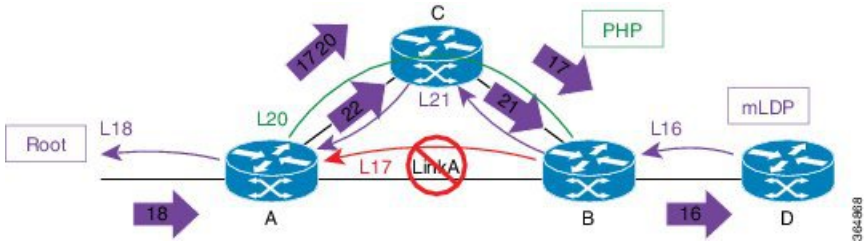
Figure 8: Link Failure



When a ink failure occurs on Link A:

1. Traffic over Link A is rerouted over the backup tunnel by imposing the traffic engineering (TE) label 20 towards mid Router C.
2. Router C performs penultimate hop popping (PHP) and removes the outer label 20.
3. Router B receives the mLDP packets with label 17 and forwards to Router D.

Figure 9: Re-optimization - Make-Before-Break



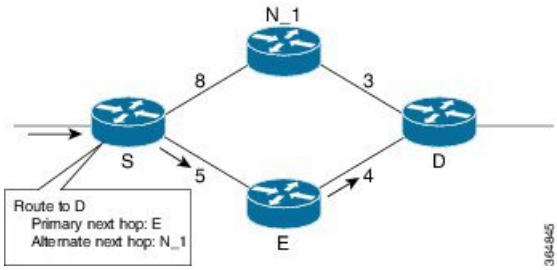
During re-optimization:

1. mLDP is notified that the root is reachable through Router C, and mLDP converges. With this, a new mLDP path is built to router A through Router C.
2. Router A forwards packets natively with old label 17 and also new label 22.
3. Router B drops traffic carried from new label 22 and forwards traffic with label 17.
4. Router B uses make-before-break (MBB) trigger to switch from either physical or bundle interface to native, label 17 to 21.
5. Router B prunes off the physical or bundle interface with a label withdraw to router A.

MLDP LFA FRR - Behavior

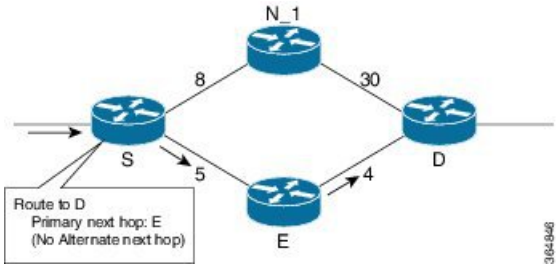
In the following scenarios, S is source router, D is the destination router, E is primary next hop, and N_1 is the alternative next hop.

Figure 10: LFA FRR Behavior - LFA Available



With LFA FRR, the source router S calculates an alternative next hop N_1 to forward traffic towards the destination router D through N_1, and installs N_1 as a the alternative next hop. On detecting the link failure between routers S and E, router S stops forwarding traffic destined for router D towards E through the failed link; instead it forwards the traffic to a pre-computed alternate next hop N_1, until a new SPF is run and the results are installed.

Figure 11: LFA FRR Behavior - LFA Not Available



In the above scenario, if the link cost between the next hop N_1 and the destination router D is increased to 30, then the next hop N_1 would no longer be a loop-free alternative. (The cost of the path, from the next hop N_1 to the destination D through the source S, would be 17, while the cost from the next hop N_1 directly to destination D would be 30). Thus, the existence of a LFA next hop is dependent on the topology and the nature of the failure, for which the alternative is calculated.

LFA Criteria

In the above example, the LFA criteria of whether N is to be the LFA next-hop is met, when:

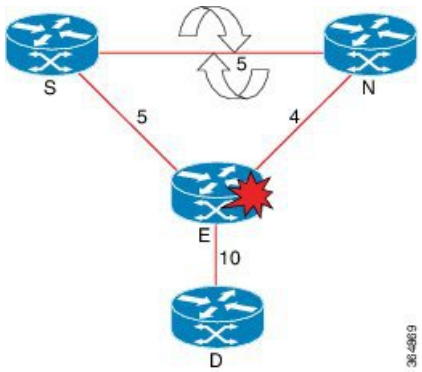
Cost of path (N_1, D) < Cost of path (N_1, S) + Cost of path (E, S) + Cost of path (D, E)

Downstream Path criteria, which is subset of LFA, is met when:

Cost of path (N_1, D) < Cost of path (E, S) + Cost of path (D, E)

Link Protecting LFA

Figure 12: Link Protecting LFA



In the above illustration, if router E fails, then both router S and router N detects a failure and switch to their alternates, causing a forwarding loop between both routers S and N. Thus, the Link Protecting LFA causes Loop on Node Failure; however, this can be avoided by using a down-stream path, which can limit the coverage of alternates. Router S will be able to use router N as a downstream alternate, however, router N cannot use S. Therefore, N would have no alternate and would discard the traffic, thus avoiding the micro-looping.

Node Protecting LFA

Link and node protecting LFA guarantees protection against either link or node failure. Depending on the protection available at the downstream node, the downstream path provides protection against a link failure; however, it does not provide protection against a node failure, thereby preventing micro looping.

The criteria for LFA selection priority is that: the Link and Node protecting LFA is greater than the Link Protecting Downstream is greater than the Link Protecting LFA.

Configure MLDP Route-Policy for Flexible Algorithm FRR

Using the MLDP route-policy option, you can enable the FRR for selected LSPs. You can enable FRR only for flexible algorithm-based LSPs, non-flexible algorithm-based LSPs, or for both. This route policy helps when you have a large number of flows and you have enabled FRR on all of them. If you have some flows that are critical and need FRR and other flows without FRR, you can apply the customization using the route policy.

If you do not configure any route policy, then the FRR is enabled on all LSPs.

The following example shows how to configure MLDP route-policy with flexible algorithm and apply the same:

```
Router#config
Router(config)#route-policy mldp-fa-frr
Router(config-rpl)#if mldp flex-algo?
<128-255>
Algorithm number
any
Any Algorithm
Router (config-rpi)#if mldp flex-algo 128 then
Router(config-rpl-if)#pass
Router(config-rpl-if)Hendif
Router(config-rpl)#if mldp flex-algo any then
Router(config-rpl-if)#pass
Router(config-rpl-if)Hendif
Router(config-rpl)Hend-policy

Router#config
```

```
Router(config)#mpls Idp mldp address-family ipv4
Router(config-ldp-mldp-af)#forwarding recursive route-policy mldp-fa-frr
Router(config-ldp-mldp-af)#
```

Configurations to Enable LFA FRR

Key Configurations To Enable LFA FRR

The key configurations to enable LFA FRR feature include:

- Router OSPF configuration

  The various configurations available under OSPF are:

  - Enabling Per-Prefix LFA
  - Excluding Interface from Using Backup
  - Adding Interfaces to LFA Candidate List
  - Restricting LFA Candidate List
  - Limiting Per-Prefix Calculation by Prefix Priority
  - Disabling Load Sharing of Backup Paths

- Router ISIS configuration
- Bidirectional Forwarding Detection (BFD) configuration
- MPLS configuration

  The various configurations available under MPLS are:

  - MBB (MLDP) configuration
  - Make Before Break (MBB) Delay X <sec> Delete Y <sec>
  - Configure FRR Timer for Scale Number of MLDP LSPs

Configuring Router OSPF LFA FRR

In OSPF configuration, configure per-prefix link based LFA to enable the LFA FRR feature. The detailed configuration steps with an example follows:

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters global configuration mode.

Step 2 **router ospf 0**

Example:

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

Enters the OSPF configuration mode.

Step 3 **area 0**

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# area 0
```

Enters the area submode under the OSPF configuration mode.

Step 4 **interface Bundle-Ether10**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar)# interface Bundle-Ether10
```

Enters the interface submode configuration, under OSPF area submode.

Step 5 **fast-reroute per-prefix**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar-if)# fast-reroute per-prefix
```

Enables the per-prefix mode of LFA calculation on the specified interface.

Step 6 **commit**

Example

Example: Configuration to Enable OSPF LFA FRR

```
!
router ospf {tag}
area {area-id}
interface {interface}
fast-reroute per-prefix enable
!
!
```

Enabling Per Prefix LFA

Lists the steps required to enable per-prefix LFA mode of LFA calculation using OSPF configuration.

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters global configuration mode.

Step 2 **router ospf 0**

Example:

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

Enters the OSPF configuration mode.

Step 3 **area 0**

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# area 0
```

Enters the area sub mode under the OSPF configuration mode.

Step 4 **interface Bundle-Ether10**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar)# interface Bundle-Ether10
```

Enters the interface sub mode configuration, under OSPF area sub mode.

Step 5 **fast-reroute per-prefix**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar-if)# fast-reroute per-prefix
```

Enables the per-prefix mode of LFA backup path calculation on the specified interface.

Step 6 **fast-reroute per-prefix remote-lfa tunnel mpls-ldp**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar-if)# fast-reroute per-prefix remote-lfa tunnel mpls-ldp
```

Enables the remote LFA on the specified interface.

Step 7 **commit**

Adding Interfaces to LFA Candidate List

Lists the steps required to add an interface to the LFA candidate list.

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters global configuration mode.

Step 2 **router ospf 0**

Example:

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

Enters the OSPF configuration mode.

Step 3 **area 0**

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# area 0
```

Enters the area submode under the OSPF configuration mode.

Step 4 **interface Bundle-Ether10**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar)# interface Bundle-Ether10
```

Enters the interface submode configuration, under OSPF area submode.

Step 5 **fast-reroute per-prefix lfa-candidate**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar-if)# fast-reroute per-prefix lfa-candidate Bundle-Ether10
```

Adds the listed interface to the LFA candidate list to compute backup paths.

Note By default, no interfaces are on the LFA candidate list.

Step 6 **commit**

Exclude Interface from Backup

Lists the steps required to exclude an interface from using backup paths for LFA calculation using OSPF configuration.

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters global configuration mode.

Step 2 **router ospf 0**

Example:

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

Enters the OSPF configuration mode.

Step 3 **area 0**

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# area 0
```

Enters the area submode under the OSPF configuration mode.

Step 4 **interface Bundle-Ether10**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar)# interface Bundle-Ether10
```

Enters the interface submode configuration, under OSPF area submode.

Step 5 **fast-reroute per-prefix exclude**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar-if)# fast-reroute per-prefix exclude Bundle-Ether10
```

Excludes the specific listed interface while calculating the LFA backup paths.

Note By default, no interfaces are excluded from the LFA backup path calculation.

Step 6 **commit**

Restricting the Backup Interfaces to the LFA Candidate List

Lists the steps required to restrict the backup interface to the LFA candidate list.

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters global configuration mode.

Step 2 **router ospf 0**

Example:

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

Enters the OSPF configuration mode.

Step 3 **area 0**

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# area 0
```

Enters the area submode under the OSPF configuration mode.

Step 4 **interface Bundle-Ether10**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar)# interface Bundle-Ether10
```

Enters the interface submode configuration, under OSPF area submode.

Step 5 **fast-reroute per-prefix use-candidate-only**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar-if)# fast-reroute per-prefix use-candidate-only
```

Restricts the calculation of the backup paths to only the interfaces listed on the LFA candidate list.

Note By default, the **fast-reroute per-prefix use-candidate-only** is disabled.

Step 6 **commit**

Limiting the Per-Prefix Calculation by Prefix-Priority

Lists the steps required to limit the per-prefix calculation by prefix-priority.

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters global configuration mode.

Step 2 **router ospf 0**

Example:

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

Enters the OSPF configuration mode.

Step 3 **area 0**

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# area 0
```

Enters the area submode under the OSPF configuration mode.

Step 4 **interface Bundle-Ether10**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar)# interface Bundle-Ether10
```

Enters the interface submode configuration, under OSPF area submode.

Step 5 **fast-reroute per-prefix prefix-limit {priority}**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar-if)# fast-reroute per-prefix prefix-limit
{priority}
```

Limits the per-prefix LFA backup path calculation by prefix-priority.

Only prefixes with the same or higher priority as specified are subjected to the per-prefix backup paths calculation.

Note By default, backup path is calculated for prefixes regardless of their priority.

Step 6 **commit**

Disabling Load Sharing of the Backup Paths

Lists the steps required to disable the load sharing of the backup paths.

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters global configuration mode.

Step 2 **router ospf 0**

Example:

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

Enters the OSPF configuration mode.

Step 3 **area 0**

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# area 0
```

Enters the area submode under the OSPF configuration mode.

Step 4 **interface Bundle-Ether10**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar)# interface Bundle-Ether10
```

Enters the interface submode configuration, under OSPF area submode.

Step 5 **fast-reroute per-prefix load-sharing disable**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar-if)# fast-reroute per-prefix load-sharing disable
```

Disables the load sharing of the backup paths.

It is used to control the load-balancing of the backup paths on a per-prefix basis.

Note By default, load-balancing of per-prefixes across all backup paths is enabled.

Step 6 **commit**

Configuring Router ISIS LFA FRR

In ISIS configuration, configure fast-reroute per-prefix to enable the LFA FRR feature.

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters the global configuration mode.

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

Example:

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

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

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

Example:

```
RP/0/RP0/CPU0:router(config-isis)# net 49.0001.0000.0000.0001.00
```

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

- Specify a NET for each routing instance if you are configuring multi-instance IS-IS.
- This example, configures a router with area ID 49.0001.0000.0000 and system ID 0000.0001.0000.0000
- To specify more than one area address, specify additional NETs. Although the area address portion of the NET differs for all of the configured items, the system ID portion of the NET must match exactly.

Step 4 **address-family ipv4 unicast**

Example:

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

Enters the address-family submode. This is supported only on unicast topologies.

Step 5 **commit**

Step 6 **interface***GigabitEthernet0/0/1/1*

Example:

```
RP/0/RP0/CPU0:router(config-isis-af)# interface GigabitEthernet0/0/1/1
```

Enters the interface submode.

Step 7 **address-family ipv4 unicast**

Example:

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

Enters the address-family submode. This is supported on unicast topologies only.

Step 8 **fast-reroute per-prefix remote-lfa tunnel mpls-ldp**

Example:

```
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix remote-lfa tunnel mpls-ldp
```

Enables LFA FRR remote LFA.

Step 9 **commit**

Configuring Bidirectional Forwarding Detection

When a local interface is down, that is, due to either a fiber cut or because of interface shutdown configuration is run, it can take a long delay in the order of tens of milliseconds for the remote peer to detect the link disconnection; so, to quickly detect the remote shut on physical port or on bundle interfaces, the physical port and bundle interfaces must be running Bidirectional Forwarding Detection (BFD) to ensure faster failure detection.

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters global configuration mode.

Step 2 **router ospf** *instance id*

Example:

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

Enters the OSPF routing configuration mode.

Step 3 **nsr**

Example:

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

Enables nonstop routing.

Step 4 **router-id** *instance id*

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# router-id 21.21.21.21
```

Specifies the router ID of the particular IPv4 address.

Step 5 **nsf** *instance name*

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# interface cisco
```

Enters the interface submode configuration, under OSPF mode.

Step 6 **address-family ipv4 unicast**

Example:

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

Enters the address-family submode. This is supported only on unicast topologies.

Step 7 **area** *instance id*

Example:

```
RP/0/RP0/CPU0:router(config-ospf-af)# area 0
```

Enters the area submode under the OSPF configuration mode.

Step 8 **bfd minimum-interval** *value*

Example:

```
RP/0/RP0/CPU0:router(config-ospf-af)# bfd minimum-interval 3
```

Sets the bidirectional forwarding detection minimum-interval value to 3.

Step 9 **bfd fast-detect**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-af)# bfd fast-detect
```

Configures bidirectional forwarding detection to fast detection.

Step 10 **bfd multiplier** *value*

Example:

```
RP/0/RP0/CPU0:router(config-ospf-af)# bfd multiplier 2
```

Configures bidirectional forwarding detection to fast detection.

Step 11 **fast-reroute per-prefix**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-af)# fast-reroute per-prefix
```

Enables the per-prefix mode of LFA calculation on the specified interface.

Step 12 **mpls traffic-eng**

Example:

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

Configures the MPLS TE under the OSPF area.

Step 13 **interface** *instance id*

Example:

```
RP/0/RP0/CPU0:router(config-ospf-af)# interface Bundle-Ether100.1
```

Configures the specified interface.

Step 14 **bfd fast-detect**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-af-if)# bfd fast-detect
```

Configures bidirectional forwarding detection to fast detection.

Step 15 **fast-reroute per-prefix**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-af-if)# fast-reroute per-prefix
```

Enables the per-prefix mode of LFA calculation on the specified interface.

Step 16 **commit**

Step 17 **interface** *instance id*

Example:

```
RP/0/RP0/CPU0:router(config-ospf-af)# interface Bundle-Ether100.1
```

Configures the specified interface.

Step 18 **bfd fast-detect**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-af-if)# bfd fast-detect
```

Configures bidirectional forwarding detection to fast detection.

Step 19 **fast-reroute per-prefix**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-af-if)# fast-reroute per-prefix
```

Enables the per-prefix mode of LFA calculation on the specified interface.

Step 20 **commit**

Step 21 **interface loopback0**

Example:

```
RP/0/RP0/CPU0:router(config)# interface loopback0
```

Example

```
router ospf 0
 nsr
 router-id 21.21.21.21
 nsf cisco
 address-family ipv4 unicast
 area 0
  bfd minimum-interval 3
  bfd fast-detect
  bfd multiplier 2
  fast-reroute per-prefix
  mpls traffic-eng
  interface Bundle-Ether100.1
    bfd fast-detect
    fast-reroute per-prefix
!
  interface Bundle-Ether100.2
    bfd fast-detect
    fast-reroute per-prefix
!
interface Loopback0
!
```

In the above configuration example, **bfd minimum-interval 3** and **bfd multiplier 2** is configured; this means, that when a core-facing interface of a remote peer is down, the router detects this disconnect event in as short a time as 6 milliseconds.

Configuring MPLS LFA FRR

In MPLS configuration, configure session protection to support LFA FRR feature. The detailed configuration steps and an example follows.

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters global configuration mode.

Step 2 **router ospf 0**

Example:

```
RP/0/RP0/CPU0:router(config)#mpls ldp
```

Enters the LDP configuration mode.

Step 3 **nsr**

Example:

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

Configures non-stop routing.

Step 4 **graceful-restart**

Example:

```
RP/0/RP0/CPU0:router(config-ldp)# graceful-restart
```

Restarts the interface.

Step 5 **router-id 20.20.20.20**

Example:

```
RP/0/RP0/CPU0:router(config-ldp)# router-id 20.20.20.20
```

Configures a router-id for the LDP process.

Step 6 **session protection**

Example:

```
RP/0/RP0/CPU0:router(config-ldp)# session protection
```

Enables LFA FRR in the per-prefix mode.

Step 7 **address-family ipv4**

Example:

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

Enters address family configuration mode.

Step 8 **forwarding recursive**

Example:

```
RP/0/RP0/CPU0:router(config-ldp)# forwarding recursive
```

Enables MLDP LFA FRR.

Step 9 **make-before-break delay 60 10**

Example:

```
RP/0/RP0/CPU0:router(config-ldp)# make-before-break delay 60 10
```

Configures make before break (MBB) delay.

Step 10 **commit**

Example

Example: Configuration to enable MLDP LFA FRR

```
mpls ldp
 nsr
 graceful-restart
 !
router-id 20.20.20.20
session protection
address-family ipv4
make-before-break delay 60 10
!
!
```

Make Before Break Configuration for LFA FRR

Make Before Break (MBB) is an inherent nature of MLDP. In MBB configuration, configure forwarding recursive to enable LFA FRR feature. If forwarding recursive is not configured, MLDP uses non-recursive method to select MLDP core facing interface towards next hop. The detailed configuration steps and an example follows.

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**mpls ldp** **Example:** `RP/0/RP0/CPU0:router(config)#mpls ldp`	Enters the LDP configuration mode.
Step 3	**log** **Example:** `RP/0/RP0/CPU0:router(config-ldp)# log`	Enters the log sub mode under the LDP sub mode.
Step 4	**neighbor** **Example:** `RP/0/RP0/CPU0:router(config-ldp-log)# neighbor`	Configures the specified neighbor to the MLDP policy.
Step 5	**nsr** **Example:** `RP/0/RP0/CPU0:router(config-ldp-log)# nsr`	Configures non-stop routing.
Step 6	**graceful-restart** **Example:** `RP/0/RP0/CPU0:router(config-ldp)# graceful-restart`	Restarts the interface.
Step 7	**commit**	
Step 8	**mldp** **Example:** `RP/0/RP0/CPU0:router(config-ldp)# mldp`	Enters the MLDP sub mode under the LDP sub mode.
Step 9	**address-family ipv4** **Example:** `RP/0/RP0/CPU0:router(config-ldp-mldp-af)# address-family ipv4`	Enters the Address Family sub mode under the MLDP sub mode.
Step 10	**forwarding recursive** **Example:**	Enables LFA FRR.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router(config-ldp-mldp-af)# forwarding recursive`	
Step 11	**make-before-break delay {seconds}** **Example:** `RP/0/RP0/CPU0:router(config-ldp-mldp-af)# make-before-break delay 60`	Sets the make-before-break delay to the specified number of seconds.
Step 12	**commit**	

Example

Example Configuration Example of MBB for LFA FRR

```
mpls ldp
log
neighbor
nsr
graceful-restart
!
mldp
address-family ipv4
forwarding recursive
make-before-break delay 60
!
!
```

Configuring Make Before Break Delay and Delete

By default, MBB is set to 10 seconds. You can configure different MBB timing to determine when the merge node starts to accept the new label.

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**mpls ldp** **Example:** `RP/0/RP0/CPU0:router(config)#mpls ldp`	Enters the LDP configuration mode.
Step 3	**mldp** **Example:** `RP/0/RP0/CPU0:router(config-ldp)# mldp`	Enters the MLDP sub mode under the LDP sub mode.
Step 4	**address-family ipv4** **Example:**	Enters the Address Family sub mode under the MLDP sub mode.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router(config-ldp-mldp)#` **`address-family ipv4`**	
Step 5	**make-before-break delay {seconds}** **Example:** `RP/0/RP0/CPU0:router(config-ldp-mldp-af)#` **`make-before-break delay 90`**	Sets the Make Before Break delay to 90 seconds.
Step 6	**make-before-break delay {seconds} delete {seconds}** **Example:** `RP/0/RP0/CPU0:router(config-ldp-mldp-af)#` **`make-before-break delay 90 delete 60`**	Sets the Make Before Break delete delay to 60 seconds.
Step 7	**commit**	

Example

Example: Make Before Break Delay And Delete

```
mldp
address-family ipv4
make-before-break delay ?
<0-600> Forwarding delay in seconds
make-before-break delay 90 ?
<0-60> Delete delay in seconds
make-before-break delay 90 delete 60
!
!
```

In the above configuration example, the MBB (delay) period is set of 90 seconds. The merge node starts accepting new label 90 seconds after detecting the link disconnection towards the head node. The delete delay is set to 60 seconds; that is, when MBB expires, the time period after which the merge node sends old label delete request to head node is 60 seconds. The default value is zero. The range of delete delay is from 30 to 60, for scale LSPs.

Overriding VRFs in IGMP Interfaces

Table 14: Feature History Table

Feature Name	Release Information	Feature Description
Support for IGMP VRF Override in Multicast Routers	Release 7.5.1	Using this feature, you can configure a multicast router interface to override the configuration specified in the local VRF table. When an IGMP client sends a join message to the multicast router, it performs a Reverse-path Forwarding (RPF) lookup for the IGMP join in the local VRF table. If the local VRF table does not have the information, the feature extends the lookup to the default (global) VRF table. This ensures that the interface in a specific VRF table is part of the outgoing list of interfaces in the global routing table for a multicast route.

All unicast traffic on the user-to-network interfaces of next-generation aggregation or core networks must be mapped to a specific VRF. They must then be mapped to an MPLS VPN on the network-to-network side. This requires the configuration of a physical interface in this specific VRF.

This feature allows mapping of IGMP packets entering through a user-to-user interface to the multicast routes in the global multicast routing table. This ensures that the interface in a specific VRF can be part of the outgoing list of interfaces in the table for a multicast route.

IGMP packets entering through a non-default VRF interface in the default (global) VRF are processed, with IGMP later distributing the interface-related multicast state (route/interface) to MRIB. This occurs through the default VRF rather than through the VRF to which the interface belongs. MRIB, PIM, MSDP, and MFIB then process the multicast state for this interface through the default VRF.

When an IGMP join for a specific (S, G) is received on the configured interface, IGMP stores this information in its VRF-specific databases. But, when sending an update to MRIB, IGMP sends this route through the default VRF. MRIB then programs this (S, G) along with this interface as an OLIST member in the default multicast routing table.

Similarly, when PIM requests information about IGMP routes from MRIB, MRIB sends this update to PIM in the context of the default VRF.

This feature specifically supports:

- Mapping of IGMP requests on an interface in a non-default VRF to the default VRF multicast routing table.
- Enabling and disabling of VRF override functionality at run time.

- Routing policy configuration at the global (default) VRF level, because routing policy configuration cannot be done at the granularity of an individual interface.
- Enablement and disablement of an IGMP VRF override on all Layer- 3 and Layer- 2 interface types, including physical Ethernet, VLAN sub-interface, bundles and VLANs over bundles.
- The same scale of multicast routes and OLIST interfaces currently supported by the platform even when VRF override functionality is operational.

Restriction

IGMP VRF Override is not supported with BVI interfaces.

Configuring IGMP VRF Override

This process consists of the following tasks:

Specifying VRF definition

SUMMARY STEPS

1. **configure**
2. **vrf** *vrf-name*
3. **address-family ipv4 unicast**
4. **import route-target 1:1**
5. **export route-target 1:1**
6. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**vrf** *vrf-name* **Example:** `RP/0/RP0/CPU0:router(config)# vrf name1`	Enters the VRF configuration sub mode.
Step 3	**address-family ipv4 unicast** **Example:** `RP/0/RP0/CPU0:router(config-vrf)# address-family ipv4 unicast`	AFI configuration for IPv4. This is supported on unicast topologies only.
Step 4	**import route-target 1:1** **Example:**	Enables VRF import.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router(config-vrf-af)# import route-target 1:1`	
Step 5	**export route-target 1:1** **Example:** `RP/0/RP0/CPU0:router(config-vrf-af)# export route-target 1:1`	Enables VRF export.
Step 6	**commit**	

Enabling Multicast Routing on default and non-default VRFs

This task enables multicast routing and forwarding on all new and existing interfaces. For the VRF override feature, multicast routing needs to be enabled on both, the default and the non-default VRFs.

SUMMARY STEPS

1. **configure**
2. **multicast-routing vrf** [*vrf-name* | *default]*
3. **interface** {*type interface-path-id* | **all**} **enable**
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**multicast-routing vrf** [*vrf-name*	*default]* **Example:** `RP/0/RP0/CPU0:router(config)# multicast-routing vrf green`	Enters multicast configuration mode for the specified VRF. Note that the default configuration mode for multicast routing is default vrf (if the non-default VRF name is not specified).
Step 3	**interface** {*type interface-path-id*	**all**} **enable** **Example:** `RP/0/RP0/CPU0:router(config-mcast-green)# interface all enable`	Enables multicast routing and forwarding on one or on all new and existing interfaces.
Step 4	**commit**	

Configuring an Interface for a Non-default VRF Instance

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **vrf** *vrf-name*
4. **ipv4 address** *address mask*
5. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**interface** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config)# interface tengige 0/1/0/0`	Enters PIM address-family IPv4 submode.
Step 3	**vrf** *vrf-name* **Example:** `RP/0/RP0/CPU0:router(config-if)# vrf name1`	Sets the VRF for the interface.
Step 4	**ipv4 address** *address mask* **Example:** `RP/0/RP0/CPU0:router(config-if)# ipv4 address 10.1.1.1 255.0.0.0`	Sets the IPv4 address for the interface.
Step 5	**commit**	

Configuring route-policy

SUMMARY STEPS

1. **configure**
2. **route-policy** *policy-name*
3. **set rpf-topology vrf default**
4. **end-policy**
5. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**route-policy** *policy-name* **Example:** `RP/0/RP0/CPU0:router(config)# route-policy policy1`	Defines a route policy.
Step 3	**set rpf-topology vrf default** **Example:** `RP/0/RP0/CPU0:router(config-rpl)# set rpf-topology vrf default`	Sets the PIM RPF topology attributes for the default VRF.
Step 4	**end-policy** **Example:** `RP/0/RP0/CPU0:router(config-rpl)# end-policy`	Ends the route-policy definition configuration.
Step 5	**commit**	

Associating a route policy to PIM configuration for the VRF receiving IGMP reports

SUMMARY STEPS

1. **configure**
2. **router pim vrf** *vrf-name* **address-family ipv4**
3. **rpf-topology route-policy** *policy-name*
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**router pim vrf** *vrf-name* **address-family ipv4**	Enters PIM address-family IPv4 submode.
Step 3	**rpf-topology route-policy** *policy-name* **Example:** `RP/0/RP0/CPU0:router(config-rpl)# rpf-topology route-policy policy1`	Associates a previously defined route-policy with the non-default VRF that receives the IGMP reports.

	Command or Action	Purpose
Step 4	commit	

Configure MVPN using Draft-Rosen (Profile 0)

Table 15: Feature History Table

Feature Name	Release Information	Feature Description
Draft-Rosen Multicast VPN (Profile 0)	Release 7.5.1	This feature is now supported on routers that have the Cisco NC57 line cards installed and operate in native and compatible modes. MVPN profile 0 uses GRE tunnels to securely transmit multicast traffic between the PE routers.
Draft-Rosen Multicast VPN (Profile 0)	Release 7.4.1	Draft-Rosen (profile 0) is a widely used MVPN model and uses GRE tunnels to securely transmit multicast traffic between the PE routers. It also enables ease of deployment by using the Protocol-Independent Multicast (PIM) protocol between edge routers (PE) and hosts (CE), and between PE routers that are running in VRF mode.

Draft-Rosen Multicast VPN (Profile 0) uses Generic Routing Encapsulation (GRE) as an overlay protocol. All multicast packets are encapsulated inside GRE. Profile 0 has PIM as the multicast routing protocol between the edge routers (PE) and hosts (CE), and between the PE routers in the VRF mode. The PE routers directly connect using a Default Multicast Distribution Tree (MDT) formed between the PE routers. The PE routers connect to each other as PIM neighbors across the Default MDT.

Benefits

- Profile 0 is a widely used model and fairly easy to deploy as Profile 0 uses the native multicast in the core and does not require any additional configuration on customers routers and in the core.

Restriction

- PIM SM is not supported in core and under VRF. Only PIM SSM is supported.
- BVI is not supported in the core.
- Auto-RP is not supported.

- If there is an IPv4 Unicast GRE tunnel configured in your network, the Maximum Transmission Unit (MTU) size of the configured Unicast GRE tunnel impacts the MTU of the Profile-0 MDT multicast. Ensure that the Profile-0 MDT multicast packet size does not exceed the MTU value of the IPv4 unicast GRE tunnel. If the multicast packet size value exceeds the MTU value of the tunnel, then the packet is dropped.
- Use the **immediate-switch** keyword only for data MDT switchover.

Supported Hardware for Draft-Rosen (Profile 0)

Draft-Rosen (Profile 0) is supported on the following hardware from IOS XR 7.5.x:

- Fixed
  - NCS-57B1
  - NCS-57C3
- Modular Linecards:
  - NC57-36H-SE
  - NC57-36H
  - NC57-24DD
  - NC57-18DD-SE

Configuration Example

Perform the following steps to configure Profile 0 on the PE devices:

```
Router# configure
Router(config)# route-policy rosen-gre
Router(config-rpl)#  set core-tree pim-default
Router(config-rpl)# end-policy

Router(config)# multicast-routing
Router(config-mcast)# vrf vpn101
Router(config-mcast-vpn101)# address-family ipv4
Router(config-mcast-vpn101-ipv4)#  mdt source Loopback0
Router(config-mcast-vpn101-ipv4)#   mdt default ipv4 232.100.0.1
Router(config-mcast-vpn101-ipv4)#   mdt data 232.101.0.1/24
Router(config-mcast-vpn101-ipv4)#   interface all enable

Router(config)# router pim
Router(config-pim)# address-family ipv4
Router(config-pim-default-ipv4)# vrf vpn101
Router(config-pim-vpn101)#  address-family ipv4
Router(config-pim-vpn101-ipv4)#   rpf topology route-policy rosen-gre
Router(config-pim-vpn101-ipv4)#   exit
Router(config-pim-vpn101-ipv4)# commit
```

Running Configuration

```
hostname PE1
logging console debugging
```

```
vrf vpn101
 vpn id 1:1
 address-family ipv4 unicast
  import route-target
   1:1
  !
  export route-target
   1:1
  !
 !
  export route-target
   1:1
  !
 !
!


route-policy rosen-gre
  set core-tree pim-default
end-policy
!
router ospf 0
 nsr
 router-id 1.1.1.1
 area 0
  interface Loopback0
  !
  interface HundredGigE0/0/0/1
  !
 !
 router bgp 100
 mvpn
 bgp router-id 1.1.1.1
 address-family ipv4 unicast
 !
 address-family vpnv4 unicast
 !
 address-family ipv4 mdt
 !
 address-family ipv4 mvpn
 !
 neighbor 2.2.2.2
  remote-as 100
  update-source Loopback0
  address-family ipv4 unicast
  !
  address-family vpnv4 unicast
  !
  address-family ipv4 mdt
  !
  address-family ipv4 mvpn
  !
 !
 neighbor 3.3.3.3
  remote-as 100
  update-source Loopback0
  address-family ipv4 unicast
  !
  address-family vpnv4 unicast
  !
  address-family ipv4 mdt
  !
  address-family ipv4 mvpn
```

```
  !
 !
 vrf vpn101
  rd 1:1
  address-family ipv4 unicast
   redistribute connected
  !
  address-family ipv4 mvpn
  !
 !
!
mpls ldp
 log
  neighbor
  nsr
  graceful-restart
 !
 mldp
 !
 interface HundredGigE0/0/0/1
 !
!
multicast-routing
 address-family ipv4
  mdt source Loopback0
  interface all enable
 !
 vrf vpn101
  address-family ipv4
   mdt source Loopback0
   interface all enable
   mdt default ipv4 232.100.0.1
   mdt data 232.101.0.1/24
  !
 !
!
router igmp
 vrf vpn101
  interface HundredGigE0/0/0/2
   static-group 232.0.0.1 1.1.10.0
  !
 !
!
router pim
 address-family ipv4
 !
 vrf vpn101
  address-family ipv4
   rpf topology route-policy rosen-gre
 !
!
```

Verification

```
Router# show pim vrf vpn101 context
Mon Jul 19 10:12:01.519 UTC

PIM context information for VRF vpn101 (0x55ed0e1bbd58)

VRF ID: 0x60000002
Table ID: 0xe0000011
Remote Table ID: 0xe0800011
MDT Default Group : 232.100.0.1
MDT Source : (1.1.1.1, Loopback0) Per-VRF
```

```
MDT Immediate Switch Not Configured
MDT handle: 0x20002e0(mdtvpn101)
Context Active, ITAL Active
Routing Enabled
Registered with MRIB
Owner of MDT Interface
Raw socket req: T, act: T, LPTS filter req: T, act: T
UDP socket req: T, act: T, UDP vbind req: T, act: T
Reg Inj socket req: T, act: T, Reg Inj LPTS filter req: T, act: T
Mhost Default Interface : HundredGigE0/0/0/0 (publish pending: F)
Remote MDT Default Group : 0.0.0.0
Backup MLC virtual interface: Null
Neighbor-filter: -
MDT Neighbor-filter: -

Router# show mrib route 232.100.0.1 detail
Mon Jul 19 10:12:01.932 UTC

IP Multicast Routing Information Base
Entry flags: L - Domain-Local Source, E - External Source to the Domain,
    C - Directly-Connected Check, S - Signal, IA - Inherit Accept,
    IF - Inherit From, D - Drop, ME - MDT Encap, EID - Encap ID,
    MD - MDT Decap, MT - MDT Threshold Crossed, MH - MDT interface handle
    CD - Conditional Decap, MPLS - MPLS Decap, EX - Extranet
    MoFE - MoFRR Enabled, MoFS - MoFRR State, MoFP - MoFRR Primary
    MoFB - MoFRR Backup, RPFID - RPF ID Set, X - VXLAN
Interface flags: F - Forward, A - Accept, IC - Internal Copy,
    NS - Negate Signal, DP - Don't Preserve, SP - Signal Present,
    II - Internal Interest, ID - Internal Disinterest, LI - Local Interest,
    LD - Local Disinterest, DI - Decapsulation Interface
    EI - Encapsulation Interface, MI - MDT Interface, LVIF - MPLS Encap,
    EX - Extranet, A2 - Secondary Accept, MT - MDT Threshold Crossed,
    MA - Data MDT Assigned, LMI - mLDP MDT Interface, TMI - P2MP-TE MDT Interface
    IRMI - IR MDT Interface, TRMI - TREE SID MDT Interface, MH - Multihome Interface

(1.1.1.1,232.100.0.1) Ver: 0xa75f RPF nbr: 1.1.1.1 Flags: RPF ME MH,
 PD: Slotmask: 0x1
     MGID: 544
  MVPN TID: 0xe0000011
  MVPN Remote TID: 0x0
  MVPN Payload: IPv4
  MDT IFH: 0x20002e0
  Up: 00:02:27
  RPF-ID: 1, Encap-ID: 0
  Incoming Interface List
    Loopback0 Flags: F A, Up: 00:02:27
  Outgoing Interface List
    Loopback0 Flags: F A, Up: 00:02:27
    HundredGigE0/0/0/1 Flags: F NS, Up: 00:02:15

(2.2.2.2,232.100.0.1) Ver: 0x8b5a RPF nbr: 1.2.1.2 Flags: RPF MD MH CD,
 PD: Slotmask: 0x1
     MGID: 545
  MVPN TID: 0xe0000011
  MVPN Remote TID: 0x0
  MVPN Payload: IPv4
  MDT IFH: 0x20002e0
  Up: 00:02:15
  RPF-ID: 1, Encap-ID: 0
  Incoming Interface List
    HundredGigE0/0/0/1 Flags: A, Up: 00:02:15
  Outgoing Interface List
    Loopback0 Flags: F NS, Up: 00:02:15
```

```
(3.3.3.3,232.100.0.1) Ver: 0xf40f RPF nbr: 1.2.1.2 Flags: RPF MD MH CD,
 PD: Slotmask: 0x1
     MGID: 546
  MVPN TID: 0xe0000011
  MVPN Remote TID: 0x0
  MVPN Payload: IPv4
  MDT IFH: 0x20002e0
  Up: 00:01:40
  RPF-ID: 1, Encap-ID: 0
  Incoming Interface List
    HundredGigE0/0/0/1 Flags: A, Up: 00:01:40
  Outgoing Interface List
    Loopback0 Flags: F NS, Up: 00:01:40
```

Multicast VPN Support based on Point to Multipoint Traffic Engineering (P2MPE)

Table 16: Feature History Table

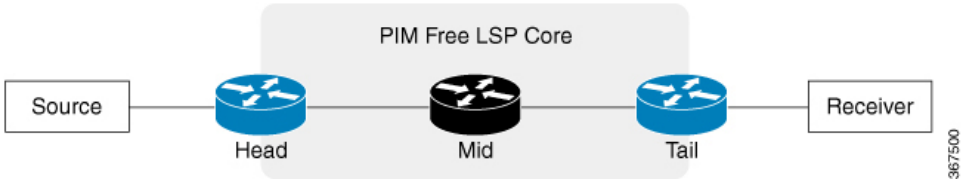
Feature Name	Release Information	Feature Description
Multicast VPN Support based on Point to Multipoint Traffic Engineering (P2MPE)	Release 7.4.1	This feature is now supported on routers that have the Cisco NC57 line cards installed and operate in native and compatibility mode.

To carry multicast traffic in service provider networks, a multicast protocol like PIM needs to be deployed to set up forwarding paths in the servicer provider core. However for an MPLS backbone network, service providers can use label encapsulation instead of IP tunneling. This approach helps to reduce the control traffic overhead on the service provider core and also leverages the MPLS traffic engineering and protection features.

The label encapsulation could be either point-to-multipoint (P2MP) label switched paths (LSPs) or multipoint-to-multipoint (MP2MP) LSPs. For creating multicast LSPs, RSVP-TE protocol extensions can be used. The RSVP-TE protocol is extended to signal P2MP LSPs across the MPLS networks. P2MP-TE feature enables transporting multicast traffic through a PIM free service provider core using P2MP-TE tunnels.

The following figure explains the topology that is used in this feature.

Figure 13: PIM Free LSP Core



In this figure, the following terminologies are used:

- Head—A router on which a TE tunnel is configured.
- Tail—The router on which the TE tunnel terminates.
- Mid—A router through which the TE tunnel passes.

A Multicast VPN (mVPN) profile is configured for the global context or per VRF. Different mVPN profiles can be applied depending on where the multicast streams need to be transported.

The following mVPN profiles are supported for the P2MP-TE feature:

- mVPN profile 8 for global context
- mVPN profile 10 for L3VPN context

Restrictions and Usage Guidelines

The following restrictions and guidelines apply for this feature:

- Only Source-Specific Multicast (SSM) traffic is supported.
- For profile 8, both IPv4 and IPv6 are supported.
- For profile 10, only IPv4 is supported.
- Fast Reroute (FRR) for P2MP-TE tunnel is not supported.
- BVI interface toward core is not supported.
- FRR for P2MP RSVP currently is not supported.
- FRR for multicast traffic with profile 14 is supported.

Configuration Example: P2MP-TE Profile 8

This example shows the P2MP-TE configuration for profile 8. You need to configure the head, mid, and tail routers in the P2MP tunnel.

The head router configuration is given as follows. This configuration includes IGP, MPLS-TE tunnel, and multicast configurations. You should also configure LDP and RSVP while configuring this feature.

The running configuration for the head router is given as follows.

The mid router only requires MPLS-TE, RSVP and an IGP like OSPF configurations. The running configuration for the mid router is given as follows:

The tail router configuration is given as follows. This configuration includes IGP, MPLS-TE tunnel and multicast configurations. Similar to head router, you should also configure RSVP and LDP while configuring this feature.

The running configuration for the tail router is given as follows:

Configuration Example: P2MP-TE Profile 10

This example shows the P2MP-TE configuration for profile 10. You need to configure the head, mid, and tail routers.

The head router configuration is given as follows. This configuration includes IGP, L3VPN, and multicast configurations. You should also configure MPLS-TE, LDP, and RSVP while configuring this feature.

The running configuration for the head router is given as follows.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# router ospf 1
RP/0/RP0/CPU0:router(config-router)# area 0
```

```
RP/0/RP0/CPU0:router(config-ospf-ar)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-ospf-ar-mpls-te)# exit
RP/0/RP0/CPU0:router(config-ospf-ar)# interface Loopback0
RP/0/RP0/CPU0:router(config-ospf-ar-if)# exit
RP/0/RP0/CPU0:router(config-ospf-ar)# interface TenGigE0/0/0/3
RP/0/RP0/CPU0:router(config-ospf-ar-if)# cost 1
RP/0/RP0/CPU0:router(config-ospf-ar-if)# network point-to-point
RP/0/RP0/CPU0:router(config-ospf-ar-if)# exit
RP/0/RP0/CPU0:router(config-ospf-ar)# exit
RP/0/RP0/CPU0:router(config-ospf)# mpls traffic-eng router-id loopback 0
RP/0/RP0/CPU0:router(config-ospf)# exit
RP/0/RP0/CPU0:router(config)# vrf vpn_2
RP/0/RP0/CPU0:router(config-vrf)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-vrf-af)# import route-target 100:2
RP/0/RP0/CPU0:router(config-vrf-af)# export route-target 120:2
RP/0/RP0/CPU0:router(config)# interface TengigE0/0/0/6
RP/0/RP0/CPU0:router(config-if)# vrf vpn_2
RP/0/RP0/CPU0:router(config-if-vrf)# ipv4 address 10.0.0.1 255.255.255.0
RP/0/RP0/CPU0:router(config)# route-policy pass-all
RP/0/RP0/CPU0:router(config)# pass
RP/0/RP0/CPU0:router(config)# end-policy
RP/0/RP0/CPU0:router(config)# router bgp 1
RP/0/RP0/CPU0:router(config-bgp)# bgp router-id 10.2.2.2
RP/0/RP0/CPU0:router(config-bgp)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-bgp-af)# address-family vpnv4 unicast
RP/0/RP0/CPU0:router(config-bgp-af)# address-family ipv4 mvpn
RP/0/RP0/CPU0:router(config-bgp)# neighbor 10.1.1.1
RP/0/RP0/CPU0:router(config-bgp-nbr)# remote-as 1
RP/0/RP0/CPU0:router(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-bgp-nbr-af)# route-policy pass-all in
RP/0/RP0/CPU0:router(config-bgp-nbr-af)# route-policy pass-all out
RP/0/RP0/CPU0:router(config-bgp-nbr)# address-family vpnv4 unicast
RP/0/RP0/CPU0:router(config-bgp-nbr-af)# route-policy pass-all in
RP/0/RP0/CPU0:router(config-bgp-nbr-af)# route-policy pass-all out
RP/0/RP0/CPU0:router(config-bgp-nbr)# address-family ipv4 mvpn
RP/0/RP0/CPU0:router(config)# multicast-routing
RP/0/RP0/CPU0:router(config-mcast)# address-family ipv4
RP/0/RP0/CPU0:router(config-mcast-default-ipv4)# interface Loopback0
RP/0/RP0/CPU0:router(config-mcast-default-ipv4-if)# enable
RP/0/RP0/CPU0:router(config-mcast-default-ipv4-if)# exit
RP/0/RP0/CPU0:router(config-mcast-default-ipv4)# mdt source Loopback0
RP/0/RP0/CPU0:router(config-mcast)# vrf vpn_2
RP/0/RP0/CPU0:router(config-mcast-vpn_2)# address-family ipv4
RP/0/RP0/CPU0:router(config-mcast-vpn_2-ipv4)# mdt source loopback0
RP/0/RP0/CPU0:router(config-mcast-vpn_2-ipv4)# core-tree-protocol rsvp-te
RP/0/RP0/CPU0:router(config-mcast-vpn_2-ipv4)# rate-per-route
RP/0/RP0/CPU0:router(config-mcast-vpn_2-ipv4)# interface all enable
RP/0/RP0/CPU0:router(config-mcast-vpn_2-ipv4)# bgp auto-discovery p2mp-te
RP/0/RP0/CPU0:router(config)# router igmp
RP/0/RP0/CPU0:router(config-igmp)# vrf vpn_2
RP/0/RP0/CPU0:router(config-igmp-vpn_2)# interface TenGigE0/0/0/6
RP/0/RP0/CPU0:router(config-igmp-vpn_2-if)# version 3
RP/0/RP0/CPU0:router(config-igmp-vpn_2-if)# exit
RP/0/RP0/CPU0:router(config)# router pim
RP/0/RP0/CPU0:router(config-pim)# vrf vpn_2
RP/0/RP0/CPU0:router(config-pim-vpn_2)# address-family ipv4
RP/0/RP0/CPU0:router(config-pim-vpn_2-ipv4)# interface TenGigE0/0/0/6
RP/0/RP0/CPU0:router(config-pim-vpn_2-ipv4-if)# enable
RP/0/RP0/CPU0:router(config)#router bgp 1
RP/0/RP0/CPU0:router(config-bgp)# bgp router-id 192.168.1.2
RP/0/RP0/CPU0:router(config-bgp)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-bgp-af)# address-family vpnv4 unicast
RP/0/RP0/CPU0:router(config-bgp-af)# address-family ipv4 mvpn
RP/0/RP0/CPU0:router(config-bgp)# neighbor 192.168.1.1
```

```
RP/0/RP0/CPU0:router(config-bgp-nbr)# remote-as 2002
RP/0/RP0/CPU0:router(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-bgp-nbr-af)# route-policy pass-all in
RP/0/RP0/CPU0:router(config-bgp-nbr-af)# route-policy pass-all out
RP/0/RP0/CPU0:router(config-bgp-nbr)# address-family vpnv4 unicast
RP/0/RP0/CPU0:router(config-bgp-nbr-af)# route-policy pass-all in
RP/0/RP0/CPU0:router(config-bgp-nbr-af)# route-policy pass-all out
RP/0/RP0/CPU0:router(config-bgp-nbr)# address-family ipv4 mvpn
RP/0/RP0/CPU0:router(config-bgp)# vrf vpn_2
RP/0/RP0/CPU0:router(config-bgp-vrf)#rd 100:2
RP/0/RP0/CPU0:router(config-bgp-vrf)#address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-bgp-vrf-af)#label mode per-prefix
RP/0/RP0/CPU0:router(config-bgp-vrf-af)#redistribute connected
RP/0/RP0/CPU0:router(config-bgp-vrf-af)#exit
RP/0/RP0/CPU0:router(config-bgp-vrf)# address-family ipv4 mvpn
```

Running configuration for the tail router is given as follows:

```
interface Loopback0
 ipv4 address 10.2.2.2 255.255.255.255
!
interface TenGigE0/0/0/3
 ipv4 address 10.3.0.2 255.255.255.0
!
router ospf 1
 area 0
  mpls traffic-eng
  interface Loopback0
  !
  interface TenGigE0/0/0/3
   cost 1
   network point-to-point
  !
!
mpls traffic-eng router-id Loopback0
!
rsvp
 interface TenGigE0/0/0/3
  bandwidth percentage 100
 !
!
mpls traffic-eng
 interface TenGigE0/0/0/3
!
mpls ldp
 discovery
  targeted-hello interval 10
 !
 router-id 10.2.2.2
 address-family ipv4
  discovery targeted-hello accept
 !
 interface TenGigE0/0/0/3
 !
! vrf vpn_2
 address-family ipv4 unicast
  import route-target
   100:2
  export route-target
   100:2

interface TenGigE0/0/0/6
 vrf vpn_2
 ipv4 address 10.6.0.2 255.255.255.0
```

```
route-policy pass-all
  pass
end-policy

router bgp 1
 bgp router-id 10.2.2.2
 address-family ipv4 unicast
 address-family vpnv4 unicast
 address-family ipv4 mvpn
 neighbor 10.1.1.1
  remote-as 1
  update-source Loopback0
  address-family ipv4 unicast
   route-policy pass-all in
   route-policy pass-all out
  address-family vpnv4 unicast
   route-policy pass-all in
   route-policy pass-all out
  address-family ipv4 mvpn
 vrf vpn_2
  rd 100:2
  address-family ipv4 unicast
   label mode per-prefix
   redistribute connected
  address-family ipv4 mvpn
!
multicast-routing
 address-family ipv4
  interface Loopback0
   enable
   !
   mdt source Loopback0
   !
  vrf vpn_2
   address-family ipv4
    mdt source Loopback0
    core-tree-protocol rsvp-te
    rate-per-route
    interface all enable
    bgp auto-discovery p2mp-te
   !
  !
router igmp
 vrf vpn_2
  interface TenGigE0/0/0/6
   version 3
 !
!
router pim
 vrf vpn_2
 address-family ipv4
   interface TenGigE0/0/0/6
     enable
   !
  !
!
```

Verification: P2MP-TE

This example shows how to verify if the multicast control state is correct on the head router using the **show mrib vrf vpn_2 route** command.

```
RP/0/RP0/CPU0:router# show mrib vrf vpn_2 route

(10.0.0.100,232.0.0.1) RPF nbr: 10.0.0.100 Flags: RPF
  Up: 00:00:38
  Incoming Interface List
    TenGigE0/0/0/0 Flags: A, Up: 00:00:38
  Outgoing Interface List
    Tunnel-mte2 Flags: F NS LI LVIF, Up: 00:00:38
```

You can also verify the multicast control state on the tail router.

```
RP/0/RP0/CPU0:router# show mrib vrf vpn_2 route

(10.0.0.100,232.0.0.1) RPF nbr: 10.1.1.1 Flags: RPF
  Up: 00:03:55
  Outgoing Interface List
    TenGigE0/0/0/6 Flags: F NS LI, Up: 00:03:55
```

This example shows how to check if the TE tunnel is established on the head router by using the **show mpls traffic-eng tunnels p2mp** command.

```
RP/0/RP0/CPU0:router# show mpls traffic-eng tunnels p2mp 2

 Name: tunnel-mte2
   Signalled-Name: head_mt2
   Status:
     Admin: up  Oper: up (Up for 00:09:37)
     Config Parameters:
      Bandwidth: 0 kbps (CT0) Priority: 7 7 Affinity: 0x0/0xffff
      Interface Bandwidth: 0 kbps
      Metric Type: TE (global)
      Fast Reroute: Not Enabled, Protection Desired: None
      Record Route: Not Enabled
      Reoptimization after affinity failure: Enabled
      Destination summary: (1 up, 0 down, 0 disabled) Affinity: 0x0/0xffff
      Auto-bw: disabled
      Destination: 10.2.2.2
        State: Up for 00:09:37
        Path options:
          path-option 1 dynamic     [active]
    Current LSP:
      lsp-id: 10002 p2mp-id: 2 tun-id: 2 src: 10.1.1.1 extid: 10.1.1.1
      LSP up for: 00:09:37 (since Fri May 25 22:32:03 UTC 2018)
      Reroute Pending: No
      Inuse Bandwidth: 0 kbps (CT0)
      Number of S2Ls: 1 connected, 0 signaling proceeding, 0 down      S2L Sub LSP:
Destination 2.2.2.2 Signaling Status: connected
        S2L up for: 00:09:37 (since Fri May 25 22:32:03 UTC 2018)
        Sub Group ID: 1 Sub Group Originator ID: 10.1.1.1
        Path option path-option 1 dynamic    (path weight 2)
        Path info (OSPF 1 area 0)
          10.0.0.5
          10.0.0.2
          10.2.2.2
    Reoptimized LSP (Install Timer Remaining 0 Seconds):
      None
    Cleaned LSP (Cleanup Timer Remaining 0 Seconds):
      None
Displayed 1 (of 101) heads, 0 (of 0) midpoints, 0 (of 0) tails
Displayed 1 up, 0 down, 0 recovering, 0 recovered heads
```

This example shows how to verify the label assignment on the head router using the **show mpls forwarding p2mp** command.

```
RP/0/RP0/CPU0:router# show mpls forwarding p2mp

Local  Outgoing    Prefix             Outgoing     Next Hop        Bytes
Label  Label       or ID              Interface                    Switched
------ ----------- ------------------ ------------ --------------- ------------
64106  64008       P2MP TE: 2         TenGigE0/0/0/2    10.0.0.5        0
```

Configure mVPN using RSVP-TE P2MP (Profile 22)

Table 17: Feature History Table

Feature Name	Release Information	
mVPN using RSVP-TE P2MP (Profile 22)	Release 7.5.1	This feature uses RSVP-TE to establish MPLS transport LSPs through traffic engineering and securely transmits multicast traffic between the PE routers in a MPLS network.

Profile 22 is a mVPN profile used for multicast traffic engineering using RSVP-TE P2MP. RSVP-TE is used to establish MPLS transport LSPs when there are traffic engineering requirements. The Default MDT consists of a full mesh of P2MP-TE tree. The P2MP tunnels are P2MP-TE auto-tunnels. PIM is used for C-Multicast signaling.

These are the characteristics of this profile:

- Dynamic P2MP-TE tunnels with BGP C-Multicast routing.
- All UMH options are supported.
- Default and Data MDT are supported.
- Customer traffic can be SSM.

Benefits

- More flexible designs per VPN.
- Proven functionality (Default-MDT and Data-MDT).
- Supports mLDP and P2MP-TE core tree protocols.
- Traffic engineering for multicast along with mVPN.

Configuration

Perform the following tasks on the edge routers (PE devices):

- Configure VRF
- Enable PIM with Profile 22

```
vrf one
 address-family ipv4 unicast
```

```
  import route-target
   1:1
  !
  export route-target
   1:1
  !
 !

router pim
 vrf one
  address-family ipv4
   rpf topology route-policy rpf-vrf-one
   mdt c-multicast-routing bgp
   interface GigabitEthernet0/0/0/1.100
    enable

route-policy rpf-vrf-one
  set core-tree p2mp-te-default
end-policy

multicast-routing
 vrf one
  address-family ipv4
   mdt source Loopback0
   mdt default p2mp-te
   rate-per-route
   interface all enable
   mdt data p2mp-te 100
   bgp auto-discovery p2mp-te
   !
   accounting per-prefix

ipv4 unnumbered mpls traffic-eng Loopback0

mpls traffic-eng
 interface GigabitEthernet0/0/0/0
 !
 interface GigabitEthernet0/0/0/2
 !
  auto-tunnel p2mp
  tunnel-id min 1000 max 2000
```

Verification

```
Router# show mrib vrf p22_20 route detail
IP Multicast Routing Information Base

Entry flags: L - Domain-Local Source, E - External Source to the Domain,
    C - Directly-Connected Check, S - Signal, IA - Inherit Accep
    IF - Inherit From, D - Drop, ME - MDT Encap, EID - Encap ID,
    MD - MDT Decap, MT - MDT Threshold Crossed, MH - MDT interface handle
    CD - Conditional Decap, MPLS - MPLS Decap, EX - Extranet
    MoFE - MoFRR Enabled, MoFS - MoFRR State, MoFP - MoFRR Primary
    MoFB - MoFRR Backup, RPFID - RPF ID Set, X - VXLAN
Interface flags: F - Forward, A - Accept, IC - Internal Copy
    NS - Negate Signal, DP - Don't Preserve, SP - Signal Present,
   II - Internal Interest, ID - Internal Disinterest, LI - Local Interest,
    LD - Local Disinterest, DI - Decapsulation Interface
   EI - Encapsulation Interface, MI - MDT Interface, LVIF - MPLS Encap,
   EX - Extranet, A2 - Secondary Accept, MT - MDT Threshold Crossed,
   MA - Data MDT Assigned, LMI - mLDP MDT Interface, TMI - P2MP-TE MDT Interface
 IRMI - IR MDT Interface, TRMI - TREE SID MDT Interface, MH - Multihome Interface
```

```
(80.0.0.11,232.1.1.1) Ver: 0xb009 RPF nbr: 80.0.0.11 Flags: RPF EID,
 PD: Slotmask: 0x0
     MGID: 592
  Up: 00:07:14
  RPF-ID: 0, Encap-ID: 262145
  Incoming Interface List
    TenGigE0/1/0/1.2 Flags: A, Up: 00:07:14
  Outgoing Interface List
    Tmdtp22/ssm/v4/vrf2 Flags: F TMI, Up: 00:07:14, Head LSM-ID: 0x4000360
```

Restrictions for MVPN Profiles

The following restriction applies to the configuration of MVPN profile:

- A router being Route Reflector (RR) and Provider Edge (PE) at that same time for BGP mVPN implementation is not supported, a type 7 and type 6 IPv4 mVPN route is not advertised by a RR, which is also a PE router, if the PE router has the VRF locally configured and when there is a local receiver.

  Use full mesh for iBGP mVPN address-family or elect any core (P) router to be the RR.

Configuration Examples for MVPN Profiles

This section provides profile-wise configuration examples for the various MVPN profiles.

Configuration Examples for Inband mLDP Profiles on Core Routers

Profile-6: VRF Inband mLDP

```
router bgp 100
 mvpn
!
multicast-routing
 vrf v61
 address-family ipv4
    mdt source Loopback0
    mdt mtu 1600
    mdt mldp in-band-signaling ipv4
    interface all enable
   !
  address-family ipv6
    mdt mtu 1600
    mdt mldp in-band-signaling ipv4
   interface all enable
 !
!
router pim
 vrf v61
  address-family ipv4
   rpf topology route-policy mldp-inband
  !
  address-family ipv6
   rpf topology route-policy mldp-inband
  !
!
route-policy mldp-inband
  set core-tree mldp-inband
```

```
end-policy
!
```

Profile-7: Global Inband mLDP

```
multicast-routing
 address-family ipv4
   mdt source Loopback0
   mdt mldp in-band-signaling ipv4
   ssm range Global-SSM-Group
  interface all enable
!
 address-family ipv6
  mdt source Loopback0
  mdt mldp in-band-signaling ipv4
  ssm range Global-SSM-Group-V6
  interface all enable
!
 router pim
  address-family ipv4
   rpf topology route-policy mldp-inband
  !
  address-family ipv6
   rpf topology route-policy mldp-inband
 !
!
route-policy mldp-inband
  set core-tree mldp-inband
end-policy
!
```