Protocol Independent Multicast - PIM

Protocol Independent Multicast (PIM) is a multicast control plane protocol that advertises multicast sources and receivers over a routed layer 3 network. Layer 3 multicast relies on PIM to advertise information about multicast capable routers, and the location of multicast senders and receivers. For this reason, multicast cannot be sent through a routed network without PIM.

PIM has two modes of operation: Sparse Mode (PIM-SM) and Dense Mode (PIM-DM).

Cumulus Linux supports only PIM Sparse Mode.

Contents

- PIM Overview
  - PIM Messages
  - PIM Neighbors
- PIM Sparse Mode (PIM-SM)
  - Any-source Multicast Routing
  - PIM Null-Register
  - PIM and ECMP
- Configure PIM
  - Configure PIM Using FRRouting
  - Example Configurations
- Source Specific Multicast Mode (SSM)
- IP Multicast Boundaries
- Multicast Source Discovery Protocol (MSDP)
- Verify PIM
  - Source Starts First
  - Receiver Joins First
- PIM in a VRF
- BFD for PIM Neighbors
- Troubleshooting
  - FHR Stuck in Registering Process
  - No *,G Is Built on LHR
  - No mroute Created on FHR
  - No S,G on RP for an Active Group
  - No mroute Entry Present in Hardware
  - Verify MSDP Session State
  - View the Active Sources
- Caveats and Errata

PIM Overview
<table>
<thead>
<tr>
<th>Network Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Hop Router (FHR)</td>
<td>The FHR is the router attached to the source. The FHR is responsible for the PIM register process.</td>
</tr>
<tr>
<td>Last Hop Router (LHR)</td>
<td>The LHR is the last router in the path, attached to an interested multicast receiver. There is a single LHR for each network subnet with an interested receiver, however multicast groups can have multiple LHRs throughout the network.</td>
</tr>
<tr>
<td>Rendezvous Point (RP)</td>
<td>The RP allows for the discovery of multicast sources and multicast receivers. The RP is responsible for sending PIM Register Stop messages to FHRs. The PIM RP address must be globally routable.</td>
</tr>
</tbody>
</table>

| PIM Shared Tree (RP Tree) or (*,G) Tree | The Shared Tree is the multicast tree rooted at the RP. When receivers want to join a multicast group, join messages are sent along the shared tree towards the RP. |
| PIM Shortest Path Tree (SPT) or (S,G) Tree | The SPT is the multicast tree rooted at the multicast source for a given group. Each multicast source has a unique SPT. The SPT can match the RP Tree, but this is not a requirement. The SPT represents the most efficient way to send multicast traffic from a source to the interested receivers. |
| Outgoing Interface (OIF) | The outgoing interface indicates the interface on which a PIM or multicast packet is be sent out. OIFs are the interfaces towards the multicast receivers. |
| Incoming Interface (IIF) | The incoming interface indicates the interface on which a multicast packet is received. An IIF can be the interface towards the source or towards the RP. |
| Reverse Path Forwarding Interface (RPF Interface) | Reverse path forwarding interface is the path used to reach the RP or source. There must be a valid PIM neighbor to determine the RPF unless directly connected to source. |
| Multicast Route (mroute) | A multicast route indicates the multicast source and multicast group as well as associated OIFs, IIFs, and RPF information. |
| Star-G mroute (*,G) | The (*,G) mroute represents the RP Tree. The * is a wildcard indicating any multicast source. The G is the multicast group. An example (*,G) is (*, 239.1.2.9). |
| S-G mroute (S,G) | This is the mroute representing the source entry. The S is the multicast source IP. The G is the multicast group. An example (S,G) is (10.1.1.1, 239.1.2.9). |

**PIM Messages**

<table>
<thead>
<tr>
<th>PIM Message</th>
<th>Description</th>
</tr>
</thead>
</table>

Do not use a spine switch as an RP. If you are running BGP on a spine switch and it is configured for allow-as in origin, BGP does not accept routes learned through other spines that do not originate on the spine itself. The RP must route to a multicast source. During a single failure scenario, this is not possible if the RP is on the spine. This also applies to Multicast Source Discovery Protocol (MSDP — see below).
PIM Hello

PIM hellos announce the presence of a multicast router on a segment. PIM hellos are sent every 30 seconds by default.

<table>
<thead>
<tr>
<th>PIM Hello Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>22.1.2.2 &gt; 224.0.0.13: PIMv2, length 34</td>
</tr>
<tr>
<td>Hello, cksum 0xfdbb (correct)</td>
</tr>
<tr>
<td>Hold Time Option (1), length 2, Value: 1m45s</td>
</tr>
<tr>
<td>0x0000: 0069</td>
</tr>
<tr>
<td>LAN Prune Delay Option (2), length 4, Value:</td>
</tr>
<tr>
<td>T-bit=0, LAN delay 500ms, Override interval 2500ms</td>
</tr>
<tr>
<td>0x0000: 01f4 09c4</td>
</tr>
<tr>
<td>DR Priority Option (19), length 4, Value: 1</td>
</tr>
<tr>
<td>0x0000: 0000 0001</td>
</tr>
<tr>
<td>Generation ID Option (20), length 4, Value: 0x2459b190</td>
</tr>
<tr>
<td>0x0000: 2459 b190</td>
</tr>
</tbody>
</table>
PIM Join/Prune (J/P)

PIM J/P messages indicate the groups that a multicast router would like to receive or no longer receive. Often PIM join/prune messages are described as distinct message types, but are actually a single PIM message with a list of groups to join and a second list of groups to leave. PIM J/P messages can be to join or prune from the SPT or RP trees (also called (*,G) joins or (S,G) joins).

PIM join/prune messages are sent to PIM neighbors on individual interfaces. Join/prune messages are never unicast.

<table>
<thead>
<tr>
<th>Protocol Protocol Independent Multicast</th>
</tr>
</thead>
<tbody>
<tr>
<td>0010 .... = Version: 2</td>
</tr>
<tr>
<td>.... 0011 = Type: Join/Prune (3)</td>
</tr>
<tr>
<td>Reserved byte(s): 00</td>
</tr>
<tr>
<td>Checksum: 0x8dd9 [correct]</td>
</tr>
<tr>
<td>PIM Options</td>
</tr>
<tr>
<td>Upstream-neighbor: 10.4.1.2</td>
</tr>
<tr>
<td>Reserved byte(s): 00</td>
</tr>
<tr>
<td>Num Groups: 1</td>
</tr>
<tr>
<td>Holdtime: 210</td>
</tr>
<tr>
<td>Group #0: 239.1.1.9/32</td>
</tr>
<tr>
<td>Num Joins: 1</td>
</tr>
<tr>
<td>IP address: 104.255.224.1/32 (SWR)</td>
</tr>
<tr>
<td>Num Prunes: 0</td>
</tr>
</tbody>
</table>

This PIM join/prune is for group 239.1.1.9, with 1 join and 0 prunes for the group. Join/prunes for multiple groups can exist in a single packet.

S,G Prune Example

| 21:49:59.470885 IP (tos 0x0, ttl 255, id 138, offset 0, flags [none], proto PIM (103), length 54) |
| 22.1.2.2 > 224.0.0.13: PIMv2, length 34 |
| Join / Prune, cksum 0xb9e5 (correct), upstream-neighbor: 22.1.2.1 |
| 1 group(s), holdtime: 3m30s |
| group #1: 225.1.0.0, joined sources: 0, pruned sources: 1 |
| pruned source #1: 33.1.1.1(S) |

PIM Register

PIM register messages are unicast packets sent from an FHR destined to the RP to advertise a multicast group. The FHR fully encapsulates the original multicast packet in PIM register messages. The RP is responsible for decapsulating the PIM register message and forwarding it along the (*,G) tree towards the receivers.

PIM Null Register

PIM null register is a special type of PIM register message where the Null-Register flag is set within the packet. Null register messages are used for an FHR to signal to an RP that a source is still sending multicast traffic. Unlike normal PIM register messages, null register messages do not encapsulate the original data packet.
PIM register stop messages are sent by an RP to the FHR to indicate that PIM register messages must no longer be sent.

### Register Stop Example

```
21:37:00.419379 IP (tos 0x0, ttl 255, id 24, offset 0, flags [none], proto PIM (103), length 38)
100.1.2.1 > 33.1.1.10: PIMv2, length 18
Register Stop, cksum 0xd8db (correct) group=225.1.0.0
source=33.1.1.1
```

IGMP membership reports are sent by multicast receivers to tell multicast routers of their interest in a specific multicast group. IGMP join messages trigger PIM *,G joins. IGMP version 2 queries are sent to the all hosts multicast address, 224.0.0.1. IGMP version 2 reports (joins) are sent to the group's multicast address. IGMP version 3 messages are sent to an IGMP v3 specific multicast address, 224.0.0.22.

IGMP leaves tell a multicast router that a multicast receiver no longer wants the multicast group. IGMP leave messages trigger PIM *,G prunes.

### IGMP Membership Report (IGMP Join)

When PIM is configured on an interface, PIM Hello messages are sent to the link local multicast group 224.0.0.13. Any other router configured with PIM on the segment that hears the PIM Hello messages build a PIM neighbor with the sending device.

PIM neighbors are stateless. No confirmation of neighbor relationship is exchanged between PIM endpoints.

### PIM Sparse Mode (PIM-SM)

PIM Sparse Mode (PIM-SM) is a **pull** multicast distribution method; multicast traffic is only sent through the network if receivers explicitly ask for it. When a receiver **pulls** multicast traffic, the network must be periodically notified that the receiver wants to continue the multicast stream.

This behavior is in contrast to PIM Dense Mode (PIM-DM), where traffic is flooded, and the network must be periodically notified that the receiver wants to stop receiving the multicast stream.

PIM-SM has three configuration options: Any-source Multicast (ASM), Bi-directional Multicast (BiDir), and Source Specific Multicast (SSM):

- **Any-source Multicast (ASM)** is the traditional, and most commonly deployed PIM implementation. ASM relies on rendezvous points to connect multicast senders and receivers that then dynamically determine the shortest path through the network between source and receiver, to efficiently send multicast traffic.
- **Bidirectional PIM (BiDir)** forwards all traffic through the multicast rendezvous point (RP) instead of tracking multicast source IPs, allowing for greater scale while resulting in inefficient forwarding of network traffic.
- **Source Specific Multicast (SSM)** requires multicast receivers to know exactly from which source they want to receive multicast traffic instead of relying on multicast rendezvous points. SSM requires the use of IGMPv3 on the multicast clients.

Cumulus Linux only supports ASM and SSM. PIM BiDir is not currently supported.

For additional information, see [RFC 7761 - Protocol Independent Multicast - Sparse Mode](https://tools.ietf.org/html/rfc7761).

### Any-source Multicast Routing

Multicast routing behaves differently depending on whether the source is sending before receivers request the multicast stream, or if a receiver tries to join a stream before there are any sources.

#### Receiver Joins First

When a receiver joins a group, an IGMP membership join message is sent to the IGMPv3 multicast group, 224.0.0.22. The PIM multicast router
for the segment that is listening to the IGMPv3 group receives the IGMP membership join message and becomes an LHR for this group.

This creates a (*,G) mroute with an OIF of the interface on which the IGMP Membership Report is received and an IIF of the RPF interface for the RP.

The LHR generates a PIM (*,G) join message and sends it from the interface towards the RP. Each multicast router between the LHR and the RP builds a (*,G) mroute with the OIF being the interface on which the PIM join message is received and an Incoming Interface of the reverse path forwarding interface for the RP.

When the RP receives the (*,G) Join message, it does not send any additional PIM join messages. The RP will maintain a (*,G) state as long as the receiver wishes to receive the multicast group.

Unlike multicast receivers, multicast sources do not send IGMP (or PIM) messages to the FHR. A multicast source begins sending, and the FHR receives the traffic and builds both a (*,G) and an (S,G) mroute. The FHR then begins the PIM register process.

**PIM Register Process**

When a first hop router (FHR) receives a multicast data packet from a source, the FHR does not know if there are any interested multicast receivers in the network. The FHR encapsulates the data packet in a unicast PIM register message. This packet is sourced from the FHR and
destined to the RP address. The RP builds an (S,G) mroute, decapsulates the multicast packet, and forwards it along the (*,G) tree.

As the unencapsulated multicast packet travels down the (*,G) tree towards the interested receivers, at the same time, the RP sends a PIM (S,G) join towards the FHR. This builds an (S,G) state on each multicast router between the RP and FHR.

When the FHR receives a PIM (S,G) join, it continues encapsulating and sending PIM register messages, but also makes a copy of the packet and sends it along the (S,G) mroute.

The RP then receives the multicast packet along the (S,G) tree and sends a PIM register stop to the FHR to end the register process.

PIM SPT Switchover

When the LHR receives the first multicast packet, it sends a PIM (S,G) join towards the FHR to efficiently forward traffic through the network. This builds the shortest path tree (SPT), or the tree that is the shortest path to the source.

When the traffic arrives over the SPT, a PIM (S,G) RPT prune is sent up the shared tree towards the RP. This removes multicast traffic from the shared tree; multicast data is only sent over the SPT.

SPT switchover can be configured on a per-group basis, allowing for some groups to never switch to a shortest path tree; this is also called SPT infinity.

The LHR now sends both (*,G) joins and (S,G) RPT prune messages towards the RP.

To configure a group to never follow the SPT, complete the following steps:
1. Create the necessary prefix-lists using the FRRouting CLI:

```bash
cumulus@switch:~$ sudo vtysh
switch# configure terminal
switch(config)# ip prefix-list spt-range permit 235.0.0.0/8 ge 32
switch(config)# ip prefix-list spt-range permit 238.0.0.0/8 ge 32
```

2. Configure SPT switchover for the `spt-range` prefix-list:

```
switch(config)# ip pim spt-switchover infinity prefix-list spt-range
```

You can view the configured prefix-list with the `net show mroute` command:

```
cumulus@switch:~$ net show mroute
Source          Group           Proto  Input      Output     TTL  Uptime
*               235.0.0.0       IGMP   swp31s0    pimreg     1    00:03:38
IGMP              br1        1    00:03:38
*               238.0.0.0       IGMP   swp31s0    br1        1    00:02:08
```

In the example above, `235.0.0.0` is configured for SPT switchover, identified by `pimreg`.

### Sender Starts Before Receivers Join

A multicast sender can send multicast data without any additional IGMP or PIM signaling. When the FHR receives the multicast traffic, it encapsulates it and sends a PIM register to the rendezvous point (RP).

When the RP receives the PIM register, it builds an (S,G) mroute; however, there is no (*,G) mroute and no interested receivers.

The RP drops the PIM register message and immediately sends a PIM register stop message to the FHR.

Receiving a PIM register stop without any associated PIM joins leaves the FHR without any outgoing interfaces. The FHR drops this multicast traffic until a PIM join is received.

PIM register messages are sourced from the interface that receives the multicast traffic and are destined to the RP address. The PIM register is not sourced from the interface towards the RP.

### PIM Null-Register

To notify the RP that multicast traffic is still flowing when the RP has no receiver, or if the RP is not on the SPT tree, the FHR periodically sends PIM null register messages. The FHR sends a PIM register with the Null-Register flag set, but without any data. This special PIM register notifies the RP that a multicast source is still sending, in case any new receivers come online.

After receiving a PIM Null-Register, the RP immediately sends a PIM register stop to acknowledge the reception of the PIM null register message.

### PIM and ECMP

PIM uses the RPF procedure to choose an upstream interface to build a forwarding state. If equal-cost multipaths (ECMP) are configured, PIM can use choose the RPF based on ECMP using hash algorithms.

The FRR `ip pim ecmp` command enables PIM to use all the available nexthops for the installation of mroutes. For example, if you have four-way ECMP, PIM spreads the S,G and *.G mroutes across the four different paths.
The `ip pim ecmp` command recalculates all stream paths in the event of a loss of path over one of the ECMP paths. Without this command, only the streams that are using the path that is lost are moved to alternate ECMP paths. Rebalance does not affect existing groups.

The `ip pim ecmp rebalance` command can cause some packet loss.

The `show ip pim nexthop` provides you with a way to review which nexthop is selected for a specific source/group:

```
cumulus@switch:~$ sudo vtysh
switch# configure terminal
switch(config)# ip pim ecmp

cumulus@switch:~$ sudo vtysh
switch# configure terminal
switch(config)# ip pim ecmp rebalance

The rebalance command can cause some packet loss.

The `show ip pim nexthop` provides you with a way to review which nexthop is selected for a specific source/group:
```
```
cumulus@switch:~$ sudo vtysh
switch# show ip pim nexthop
Number of registered addresses: 3
Address          Interface      Nexthop          
---------------------------
6.0.0.9           swp31s0        169.254.0.9
6.0.0.9           swp31s1        169.254.0.25
6.0.0.11          lo             0.0.0.0
6.0.0.10          swp31s0        169.254.0.9
6.0.0.10          swp31s1        169.254.0.25
```

Configure PIM

To configure PIM using NCLU:

1. Configure the PIM interface:
   ```
cumulus@switch:~$ net add interface swp1 pim sm
```

PIM must be enabled on all interfaces facing multicast sources or multicast receivers, as well as on the interface where the RP address is configured.

2. Optional: Run the following command to enable IGMP (either version 2 or 3) on the interfaces with hosts attached. IGMP version 3 is the default, so you only need to specify the version if you want to use IGMP version 2:
   ```
cumulus@switch:~$ net add interface swp1 igmp version 2
```

You must configure IGMP on all interfaces where multicast receivers exist.
3. Configure a group mapping for a static RP:

```
cumulus@switch:~$ net add pim rp 192.168.0.1
```

Unless you are using PIM SSM, each PIM-SM enabled device must configure a static RP to a group mapping, and all PIM-SM enabled devices must have the same RP to group mapping configuration.

IP PIM RP group ranges can overlap. Cumulus Linux performs a longest prefix match (LPM) to determine the RP. For example:

```
cumulus@switch:~$ net add pim rp 192.168.0.1 224.10.0.0/16
cumulus@switch:~$ net add pim rp 192.168.0.2 224.10.2.0/24
```

In this example, if the group is in 224.10.2.5, the RP that gets selected is 192.168.0.2. If the group is 224.10.15, the RP that gets selected is 192.168.0.1.

4. Review and commit your changes:

```
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Configure PIM Using FRRouting

PIM is included in the FRRouting package. For proper PIM operation, PIM depends on Zebra. PIM relies on unicast routing to be configured and operational to do RPF operations. Therefore, you must configure some other routing protocol or static routes.

To configure PIM on a switch using FRR:

1. Open the `/etc/frr/daemons` file in a text editor.
2. Add the following line to the end of the file to enable `pimd`, then save the file:

```
zebra=yes
pimd=yes
```
3. Run the `systemctl restart` command to restart FRRouting:

```
cumulus@switch:~$ sudo systemctl restart frr
```
4. In a terminal, run the `vtysh` command to start the FRRouting CLI on the switch.

```
cumulus@switch:~$ sudo vtysh
cumulus#
```
5. Run the following commands to configure the PIM interfaces:

```
cumulus# configure terminal
cumulus(config)# int swp1
cumulus(config-if)# ip pim sm
```
PIM must be enabled on all interfaces facing multicast sources or multicast receivers, as well as on the interface where the RP address is configured.

6. **Optional**: Run the following commands to enable IGMP (either version 2 or 3) on the interfaces with hosts attached. IGMP version 3 is the default; you only need to specify the version if you want to use IGMP version 2:

```
cumulus# configure terminal
cumulus(config)# int swp1
cumulus(config-if)# ip igmp
cumulus(config-if)# ip igmp version 2 #skip this step if you are using version 3
```

You must configure IGMP on all interfaces where multicast receivers exist.

7. Configure a group mapping for a static RP:

```
cumulus# configure terminal
cumulus(config)# ip pim rp 192.168.0.1
```

Each PIM-SM enabled device must configure a static RP to a group mapping, and all PIM-SM enabled devices must have the same RP to group mapping configuration.

IP PIM RP group ranges can overlap. Cumulus Linux performs a longest prefix match (LPM) to determine the RP. For example:

```
cumulus(config)# ip pim rp 10.0.0.13 224.10.0.0/16
```

**Example Configurations**

- **Complete Multicast Network Configuration Example**
  - The following is example configuration:
RP Configuration

RP# show run
Building configuration...
Current configuration:
!
log syslog
ip multicast-routing
ip pim rp 192.168.0.1 224.0.0.0/4
username cumulus nopassword
!
!
interface lo
  description RP Address interface
  ip ospf area 0.0.0.0
  ip pim sm
!
interface swp1
  description interface to FHR
  ip ospf area 0.0.0.0
  ip ospf network point-to-point
  ip pim sm
!
interface swp2
  description interface to LHR
  ip ospf area 0.0.0.0
  ip ospf network point-to-point
  ip pim sm
!
router ospf
  ospf router-id 192.168.0.1
!
line vty
!
end
FHR Configuration

FHR# show run
!
log syslog
ip multicast-routing
ip pim rp 192.168.0.1 224.0.0.0/4
username cumulus nopassword
!
interface bridge10.1
    description Interface to multicast source
    ip ospf area 0.0.0.0
    ip ospf network point-to-point
    ip pim sm
!
interface lo
    ip ospf area 0.0.0.0
    ip pim sm
!
interface swp49
    description interface to RP
    ip ospf area 0.0.0.0
    ip ospf network point-to-point
    ip pim sm
!
interface swp50
    description interface to LHR
    ip ospf area 0.0.0.0
    ip ospf network point-to-point
    ip pim sm
!
router ospf
    ospf router-id 192.168.1.1
!
line vty
!
end
LHR Configuration

LHR# show run
!
log syslog
ip multicast-routing
ip pim rp 192.168.0.1 224.0.0.0/4
username cumulus nopassword
!
interface bridge10.1
  description interface to multicast receivers
  ip igmp
  ip ospf area 0.0.0.0
  ip ospf network point-to-point
  ip pim sm
!
interface lo
  ip ospf area 0.0.0.0
  ip pim sm
!
interface swp49
  description interface to RP
  ip ospf area 0.0.0.0
  ip ospf network point-to-point
  ip pim sm
!
interface swp50
  description interface to FHR
  ip ospf area 0.0.0.0
  ip ospf network point-to-point
  ip pim sm
!
router ospf
  ospf router-id 192.168.2.2
!
line vty
!
end

Source Specific Multicast Mode (SSM)

The source-specific multicast method uses prefix-lists to configure a receiver to only allow traffic to a multicast address from a single source. This removes the need for an RP, as the source must be known before traffic can be accepted. The default range is 232.0.0.0/8, and must be further configured by setting a prefix-list.

The example process below configures a prefix-list named ssm-range, and prefix-lists permitting traffic from 230.0.0.0/8 and 238.0.0.0/8, for prefixes longer than 32.

PIM considers 232.0.0.0/8 the default range if the ssm range is not configured. If this default is overridden with a prefix-list, all ranges that should be considered must be in the prefix-list.
You can also perform this configuration with the FRRouting CLI:

```plaintext
cumulus@switch:~$ sudo vtysh
switch# conf t
switch(config)# ip prefix-list ssm-range seq 5 permit 232.0.0.0/8 ge 32
switch(config)# ip prefix-list ssm-range seq 10 permit 238.0.0.0/8 ge 32
switch(config)# ip pim ssm prefix-list ssm-range
switch(config)# exit
switch# write mem
```

To view the existing prefix-lists, run the `net show ip` command:

```plaintext
cumulus@switch:~S net show ip prefix-list ssm-range
ZEBRA: ip prefix-list ssm-range: 2 entries
  seq 5 permit 232.0.0.0/8 ge 32
  seq 10 permit 238.0.0.0/8 ge 32
OSPF: ip prefix-list ssm-range: 2 entries
  seq 5 permit 232.0.0.0/8 ge 32
  seq 10 permit 238.0.0.0/8 ge 32
PIM: ip prefix-list ssm-range: 2 entries
  seq 5 permit 232.0.0.0/8 ge 32
  seq 10 permit 238.0.0.0/8 ge 32
```

**IP Multicast Boundaries**

Multicast boundaries enable you to limit the distribution of multicast traffic by setting boundaries with the goal of pushing multicast to a subset of the network.

With such boundaries in place, any incoming IGMP or PIM joins are dropped or accepted based upon the prefix-list specified. The boundary is implemented by applying an IP multicast boundary OIL (outgoing interface list) on an interface.

To configure the boundary, use NCLU:

1. Create a prefix-list as above.
2. Configure the IP multicast boundary:

```plaintext
cumulus@switch:~$ net add <interface> multicast boundary oil <prefix-list>
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```
Multicast Source Discovery Protocol (MSDP)

You can use the Multicast Source Discovery Protocol (MSDP) to connect multiple PIM-SM multicast domains together, using the PIM-SM RPs. By configuring any cast RPs with the same IP address on multiple multicast switches (primarily on the loopback interface), the PIM-SM limitation of only one RP per multicast group is relaxed. This allows for an increase in both failover and load-balancing throughout.

When an RP discovers a new source (typically a PIM-SM register message), a source-active (SA) message is sent to each MSDP peer. The peer then determines if any receivers are interested.

The following steps demonstrate how to configure a Cumulus switch to use the MSDP:

1. Add an anycast IP address to the loopback interface for each RP in the domain:
   ```
   cumulus@rp01:~$ net add loopback lo ip address 10.1.1.1/32
   cumulus@rp01:~$ net add loopback lo ip address 10.1.1.100/32
   ```

2. On every multicast switch, configure the group to RP mapping using the anycast address:
   ```
   cumulus@switch:$ net add pim rp 100.1.1.100 224.0.0.0/4
   cumulus@switch:$ net pending
   cumulus@switch:$ net commit
   ```

3. Configure the MSDP mesh group for all active RPs (the following example uses 3 RPs):
   ```
   cumulus@rp01:$ net add msdp mesh-group cumulus member 100.1.1.2
   cumulus@rp01:$ net add msdp mesh-group cumulus member 100.1.1.3
   cumulus@rp02:$ net add msdp mesh-group cumulus member 100.1.1.1
   cumulus@rp02:$ net add msdp mesh-group cumulus member 100.1.1.3
   cumulus@rp03:$ net add msdp mesh-group cumulus member 100.1.1.1
   cumulus@rp03:$ net add msdp mesh-group cumulus member 100.1.1.2
   ```

4. Pick the local loopback address as the source of the MSDP control packets:
   ```
   cumulus@rp01:$ net add msdp mesh-group cumulus source 100.1.1.1
   ```

Cumulus Linux MSDP support is primarily for anycast-RP configuration, rather than multiple multicast domains. You must configure each MSDP peer in a full mesh, as SA messages are not received and re-forwarded.

Cumulus Linux currently only supports one MSDP mesh-group.

The mesh group must include all RPs in the domain as members, with a unique address as the source. This configuration results in MSDP peerings between all RPs.
5. Inject the anycast IP address into the IGP of the domain.

```plaintext
cumulus@rp02:$ net add msdp mesh-group cumulus source 100.1.1.2
```

```plaintext
cumulus@rp03:$ net add msdp mesh-group cumulus source 100.1.1.3
```

If the network is unnumbered and uses unnumbered BGP as the IGP, avoid using the anycast IP address for establishing unicast or multicast peerings. For PIM-SM, ensure that the unique address is used as the PIM hello source by setting the source:

```plaintext
cumulus@rp01:$ net add interface lo pim use-source 100.1.1.1
```

**Verify PIM**

The following outputs are based on the Cumulus Reference Topology with cldemo-pim.

**Source Starts First**

On the FHR, an mroute is built, but the upstream state is *Prune*. The FHR flag is set on the interface receiving multicast.

Use the `net show mroute` command (or `show ip mroute` in FRR) to review detailed output for the FHR:
On the RP, no mrout state is created, but the `net show pim upstream` output includes the S,G:

```
cumulus@rp01:$ net show mroute
Source  Group     Proto  Input   Output   TTL  Uptime
172.16.5.105 239.1.1.1 none br0    none    0   --:--:--
!
cumulus@rp01:$ net show pim upstream
IIf  Source          Group    State      Uptime  JoinTimer        RSTimer          KATimer    RefCnt
swp30 172.16.5.105    239.1.1.1 Prune 00:00:19 --:--:-- 00:00:19 --:--:-- 00:02:46 1
!
```

As a receiver joins the group, the mrout output interface on the FHR transitions from `none` to the RPF interface of the RP:

```
cumulus@fhr:$ net show mroute
Source  Group     Proto  Input   Output   TTL  Uptime
172.16.5.105 239.1.1.1 none br0    none    0   --:--:--
!
```
### cumulus@fhr:~$ net show mroute

<table>
<thead>
<tr>
<th>Source</th>
<th>Group</th>
<th>Proto</th>
<th>Input</th>
<th>Output</th>
<th>TTL</th>
<th>Uptime</th>
</tr>
</thead>
<tbody>
<tr>
<td>172.16.5.105</td>
<td>239.1.1.1</td>
<td>PIM</td>
<td>br0</td>
<td>swp1</td>
<td>1</td>
<td>00:05:40</td>
</tr>
</tbody>
</table>

### cumulus@fhr:~$ net show pim upstream

<table>
<thead>
<tr>
<th>Iif</th>
<th>Source</th>
<th>Group</th>
<th>State</th>
<th>Uptime</th>
<th>JoinTimer</th>
</tr>
</thead>
<tbody>
<tr>
<td>br0</td>
<td>172.16.5.105</td>
<td>239.1.1.1</td>
<td>Prune</td>
<td>00:48:23</td>
<td>--:--:--</td>
</tr>
</tbody>
</table>

### cumulus@fhr:~$ net show pim upstream-join-desired

<table>
<thead>
<tr>
<th>Interface</th>
<th>Source</th>
<th>Group</th>
<th>LostAssert</th>
<th>Joins</th>
<th>PimInclude</th>
<th>EvalJD</th>
</tr>
</thead>
<tbody>
<tr>
<td>swp51</td>
<td>172.16.5.105</td>
<td>239.1.1.1</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
</tbody>
</table>

### cumulus@fhr:~$ net show pim state

<table>
<thead>
<tr>
<th>Source</th>
<th>Group</th>
<th>IIF</th>
<th>OIL</th>
</tr>
</thead>
<tbody>
<tr>
<td>172.16.5.105</td>
<td>239.1.1.1</td>
<td>br0</td>
<td>swp1</td>
</tr>
</tbody>
</table>

### cumulus@rp01:~$ net show mroute

<table>
<thead>
<tr>
<th>Source</th>
<th>Group</th>
<th>Proto</th>
<th>Input</th>
<th>Output</th>
<th>TTL</th>
<th>Uptime</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>239.1.1.1</td>
<td>PIM</td>
<td>lo</td>
<td>swp1</td>
<td>1</td>
<td>00:09:59</td>
</tr>
<tr>
<td>172.16.5.105</td>
<td>239.1.1.1</td>
<td>PIM</td>
<td>swp30</td>
<td>swp1</td>
<td>1</td>
<td>00:09:59</td>
</tr>
</tbody>
</table>

### cumulus@rp01:~$ net show pim upstream

<table>
<thead>
<tr>
<th>Iif</th>
<th>Source</th>
<th>Group</th>
<th>State</th>
<th>Uptime</th>
<th>JoinTimer</th>
</tr>
</thead>
<tbody>
<tr>
<td>lo</td>
<td>*</td>
<td>239.1.1.1</td>
<td>Joined</td>
<td>00:10:01</td>
<td>00:00:59</td>
</tr>
<tr>
<td>swp30</td>
<td>172.16.5.105</td>
<td>239.1.1.1</td>
<td>Joined</td>
<td>00:00:01</td>
<td>00:00:59</td>
</tr>
</tbody>
</table>

### cumulus@rp01:~$ net show pim upstream-join-desired

<table>
<thead>
<tr>
<th>Interface</th>
<th>Source</th>
<th>Group</th>
<th>LostAssert</th>
<th>Joins</th>
<th>PimInclude</th>
<th>EvalJD</th>
</tr>
</thead>
<tbody>
<tr>
<td>swp1</td>
<td>*</td>
<td>239.1.1.1</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
</tbody>
</table>

### cumulus@rp01:~$ net show pim state

<table>
<thead>
<tr>
<th>Source</th>
<th>Group</th>
<th>IIF</th>
<th>OIL</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>239.1.1.1</td>
<td>lo</td>
<td>swp1</td>
</tr>
<tr>
<td>172.16.5.105</td>
<td>239.1.1.1</td>
<td>swp30</td>
<td>swp1</td>
</tr>
</tbody>
</table>

**Receiver Joins First**

On the LHR attached to the receiver:
On the RP:
PIM in a VRF

VRFs divide the routing table on a per-tenant basis, ultimately providing for separate layer 3 networks over a single layer 3 infrastructure. With a VRF, each tenant has its own virtualized layer 3 network, so IP addresses can overlap between tenants.

PIM in a VRF enables PIM trees and multicast data traffic to run inside a layer 3 virtualized network, with a separate tree per domain or tenant. Each VRF has its own multicast tree with its own RP(s), sources, and so on. Therefore, you can have one tenant per corporate division, client, or product; for example.

VRFs on different switches typically connect or are peered over subinterfaces, where each subinterface is in its own VRF, provided MP-BGP VPN is not enabled or supported.

To configure PIM in a VRF, run the following commands. First, add the VRFs and associate them with switch ports:

cumulus@rp01:~$ net add vrf blue
cumulus@rp01:~$ net add vrf purple
cumulus@rp01:~$ net add interface swp1 vrf blue
cumulus@rp01:~$ net add interface swp2 vrf purple

Then add the PIM configuration to FRR, review and commit the changes:
These commands create the following configuration in the `/etc/network/interfaces` file and the `/etc/frr/frr.conf` file:

```
auto purple
iface purple
  vrf-table auto

auto blue
iface blue
  vrf-table auto

auto swp1
iface swp1
  vrf purple

auto swp49.1
iface swp49.1
  vrf purple

auto swp2
iface swp2
  vrf blue

auto swp49.2
iface swp49.2
  vrf blue

...
ip pim rp 192.168.0.1 224.0.0.0/4

vrf purple
    ip pim rp 192.168.0.1 224.0.0.0/4
!
vrf blue
    ip pim rp 192.168.0.1 224.0.0.0/4
!

int swp1 vrf purple
    ip pim sm
    ip igmp version 2

int swp2 vrf blue
    ip pim sm
    ip igmp version 3

int swp49.1 vrf purple
    ip pim sm

int swp49.2
    ip pim sm

router bgp 65000 vrf purple
    Bgp router-id 10.1.1.1
    Neighbor PURPLE peer-group
    Neighbor PURPLE remote-as external
    neighbor swp49.1 interface peer-group PURPLE

router bgp 65001 vrf blue
    bgp router-id 10.1.1.2
    neighbor BLUE peer-group
    neighbor BLUE remote-as external
    neighbor swp49.2 interface peer-group BLUE

In FRR, you can use show commands to display VRF information:

```
net show mroute vrf blue
Source     Group     Proto Input   Output   TTL Uptime
11.1.0.1    239.1.1.1 IGMP  swp32s0  swp32s1 1  00:01:13
            IGMP      br0.200  1  00:01:13
*                   IGMP  mars  pimreg1001 1  00:01:13
            IGMP  swp32s1  1  00:01:12
            IGMP      br0.200  1  00:01:13
```
BFD for PIM Neighbors

You can use bidirectional forward detection (BFD) for PIM neighbors to quickly detect link failures. When you configure an interface, include the `pim bfd` option:

```sh
cumulus@switch:~$ net add interface swp31s3 pim bfd
cumulus@switch:~$ net pending
```

Troubleshooting

FHR Stuck in Registering Process

When a multicast source starts, the FHR sends unicast PIM register messages from the RPF interface towards the source. After the PIM register is received by the RP, a PIM register stop message is sent from the RP to the FHR to end the register process. If an issue occurs with this communication, the FHR becomes stuck in the registering process, which can result in high CPU, as PIM register packets are generated by the FHR CPU and sent to the RP CPU.

To assess this issue:

- Review the FHR. The output interface of `pimreg` can be seen here. If this does not change to an interface within a few seconds, the FHR is likely stuck.

To troubleshoot the issue:

1. Validate that the FHR can reach the RP. If the RP and FHR can not communicate, the registration process fails:

   ```sh
   cumulus@fhr:~$ ping 10.0.0.21 -I br0
   PING 10.0.0.21 (10.0.0.21) from 172.16.5.1 br0: 56(84) bytes of data.
   ^C
   4 packets transmitted, 0 received, 100% packet loss, time 3000ms
   ```

2. On the RP, use `tcpdump` to see if the PIM register packets are arriving:
3. If PIM registration packets are being received, verify that they are seen by PIM by issuing `debug pim packets` from within FRRouting:

```
cumulus@fhr:~$ sudo vtysh -c "debug pim packets"
PIM Packet debugging is on
```

```
cumulus@rp01:~$ sudo tcpdump -i swp30
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on swp30, link-type EN10MB (Ethernet), capture size 262144 bytes
23:33:17.524982 IP 172.16.5.1 > 10.0.0.21: PIMv2, Register, length 66
```

4. Repeat the process on the FHR to see if PIM register stop messages are being received on the FHR and passed to the PIM process:

```
cumulus@fhr:~$ sudo tcpdump -i swp51
23:58:59.841625 IP 172.16.5.1 > 10.0.0.21: PIMv2, Register, length 28
23:58:59.842466 IP 10.0.0.21 > 172.16.5.1: PIMv2, Register Stop, length 18
```

```
cumulus@fhr:~$ sudo vtysh -c "debug pim packets"
PIM Packet debugging is on
```

```
cumulus@fhr:~$ sudo tail -f /var/log/frr/frr.log
2016/10/19 23:59:38 PIM: Recv PIM REGSTOP packet from 10.0.0.21 to 172.16.5.1 on swp51: ttl=255 pim_version=2 pim_msg_size=18 checksum=5a39
```

No *,G Is Built on LHR

The most common reason for a *,G to not be built on an LHR is for if both PIM and IGMP are not enabled on an interface facing a receiver.

```
lhr# show run
!
interface br0
ip igmp
ip ospf area 0.0.0.0
ip pim sm
```

To troubleshoot this issue, if both PIM and IGMP are enabled, ensure that IGMPv3 joins are being sent by the receiver:
No mroute Created on FHR

To troubleshoot this issue:

1. Verify that multicast traffic is being received:

   ```
cumulus@fhr:~$ sudo tcpdump -i br0
   tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
   listening on br0, link-type EN10MB (Ethernet), capture size 262144 bytes
   00:11:52.944745 IP 172.16.5.105.51570 > 239.2.2.9.1000: UDP, length 9
   ```

2. Verify that PIM is configured on the interface facing the source:

   ```
fhr# show run
   !
   interface br0
   ip ospf area 0.0.0.0
   ip pim sm
   ```

3. If PIM is configured, verify that the RPF interface for the source matches the interface on which the multicast traffic is received:

   ```
fhr# show ip rpf 172.16.5.105
   Routing entry for 172.16.5.0/24 using Multicast RIB
   Known via "connected", distance 0, metric 0, best
   * directly connected, br0
   ```

4. Verify that an RP is configured for the multicast group:

   ```
fhr# show ip pim rp-info
   RP address   group/prefix-list   OIF      I am RP
   10.0.0.21     224.0.0.0/4        swp51    no
   ```

No S,G on RP for an Active Group

An RP does not build an mroute when there are no active receivers for a multicast group, even though the mroute was created on the FHR:
```
cumulus@rp01:~$ net show mroute
Source          Group           Proto  Input      Output     TTL  Uptime
spine01#

cumulus@rp01:~$ net show mroute
Source          Group           Proto  Input      Output     TTL  Uptime
172.16.5.105    239.2.2.9       none   br0        none       0    --:--:--
```

This is expected behavior. You can see the active source on the RP with the `show ip pim upstream` command:

```
cumulus@rp01:~$ net show pim upstream
Iif       Source          Group           State       Uptime   JoinTimer
RSTimer   KATimer   RefCnt
swp30     172.16.5.105    239.2.2.9       Prune       00:08:03 --:--:--  00:02:20       1
!  
cumulus@rp01:~$ net show mroute
Source          Group           Proto  Input      Output     TTL  Uptime
```

**No mroute Entry Present in Hardware**

Use the `cl-resource-query` command to verify that the hardware IP multicast entry is the maximum value:

```
cumulus@switch:~$ cl-resource-query | grep Mcast
Total Mcast Routes:        450,     0% of maximum value    450
```

For Mellanox chipsets, refer to [TCAM Resource Profiles for Mellanox Switches](#).

**Verify MSDP Session State**

Run the following commands to verify the state of MSDP sessions:

```
cumulus@switch:~$ net show msdp mesh-group
Mesh group : pod1
    Source : 100.1.1.1
    Member          State
    100.1.1.2       established
    100.1.1.3       established

cumulus@switch:~$
cumulus@switch:~$ net show msdp peer
Peer          Local        State    Uptime   SaCnt
100.1.1.2              100.1.1.1  established  00:07:21       0
100.1.1.3              100.1.1.1  established  00:07:21       0
```

**View the Active Sources**
Review the active sources learned locally (through PIM registers) and from MSDP peers:

```
cumulus@switch:~$ net show msdp sa
Source     Group          RP   Local  SPT  Uptime
44.1.11.2  239.1.1.1      100.1.1.1 n  n  00:00:40
44.1.11.2  239.1.1.2      100.1.1.1 n  n  00:00:25
```

Caveats and Errata

- Cumulus Linux only supports PIM sparse mode (PIM-SM), any-source multicast (PIM-SM ASM), and source-specific multicast (SSM). Dense mode and bidirectional multicast are not supported.
- Non-native forwarding (register decapsulation) is not supported. Initial packet loss is expected while the PIM *,G tree is built from the rendezvous point to the FHR to trigger native forwarding.
- Cumulus Linux does not currently build an S,G mroute when forwarding over an *,G tree.