Protocol Independent Multicast (pim)
Internet-Draft
Intended status: Standards Track
Expires: September 10, 2015
March 9, 2015

A. Peter, Ed.
R. Kebler
V. Nagarajan
Juniper Networks, Inc.

Reliable Transport For PIM Register States
draft-anish-reliable-pim-registers-00

Abstract

This document introduces a hard-state, reliable transport for the existing PIM-SM registers states. This eliminates the needs for periodic NULL-registers and register-stop in response to each data-register or NULL-registers.

This specification uses the existing PIM reliability mechanisms defined by PIM Over Reliable Transport [RFC6559]. This is simply a means to transmit reliable PIM messages and does not require the support for Join/Prune messages over PORT as defined in [RFC6559].

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC2119 [RFC2119].

Status of This Memo

Table of Contents

1. Introduction

Protocol Independent Multicast-Sparse Mode Register mechanism serves the following purposes.

a, With a register, First-Hop-Router (FHR) informs the RP (that way the network) that a particular multicast stream is active

b, A register helps avoid initial packet loss. (Initial packet loss could happen in an anycast-RP deployment even when packet registers are used.)

c, Throught its periodic refreshes register keeps RP informed about the aliveness of this multicast stream.

As it is defined in [RFC4601] , register mechanisms face limitations, when the number of multicast streams on the network is high, especially when one RP is expected to serve a large number of streams. These problems are mainly due to these factors.

a, PIM register needs control-plane and data-plane intervention to handle it.

b, Due to the nature of PIM register, First-Hop-Router and RP now needs to maintain states and timers for each register state entry.

c, PIM register's requirements for periodic refresh and expiry, is quite aggressive and makes them vulnerable when the PIM speaker could not find cycles to meet these needs

2. Reliable Register Overview

Reliable PIM register extends PIM PORT [RFC6559] to have PIM register states to be sent over a reliable transport.

This document introduces 'targeted' hellos between any two PIM peers. This helps in capability negotiation and discovery between two PIM speakers (FHR and RP in the context of this document). Once this discovery happens, First-Hop-Router would setup a reliable transport connection based on the negotiated parameters.

Over this reliable connection, First-Hop-Router would start sending to RP the source and group addresses of the multicast streams active with it. When any of this stream stops, First-Hop-Router would sent an update to RP about the streams that have stopped. This way once a reliable connection is setup, First-Hop-Router would update RP with its existing active multicast streams. Subsequently it would sent incremental updates about the change to RP.

For a multicast application that may demand initial packets or for bursty sources existing data-registers may be used. For them the RP would now respond with a 'reliable'-register-stop, which could persist until the First-Hop-Router withdraws the register-state.

3. Targeted Hellos

PIM hellos defined in PIM-SM [RFC4601] confines them to link level. This document extends these hellos to support 'targeted' hellos.

Targeted hellos are identical to existing hellos messages except that they would have an unicast address as its destination address. It would traverse multiple hops using the unicast routing to reach the targeted hello neighbor.

3.1. New Hello Optional TLV's

Option Type: Targeted hello

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Type = 36 (suggested)     |         Length = 4            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |F|R|                        Reserved                 |  Exp  |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: PIM Hello Optional TLV

Assigned Hello Type values can be found in IANA PIM registry.

Type: This is subject to IANA allocation. Suggested is 36 which is the next unused type.

Length: Length in bytes for the value part of the Type/Length/Value encoding fixed as 4.

F: To be set by a router that wants to be a First-Hop-Router.

R: To be set by a RP that is capable taking the role of an RP as per the current states.

Reserved: Set to zero on transmission and ignored on receipt.

Exp: For experimental use [RFC3692]. One expected use of these bits would be to signal experimental capabilities. For example, if a router supports an experimental feature, it may set a bit to indicate this. The default behavior, unless a router supports a particular experiment, is to keep these bits reset and ignore the bits on receipt.

3.2. Differences from Link-Level hellos

The Major differences that Link-Level-Hellos have over Interface hellos are,

1. Destination address would be an unicast address unlike ALL-PIM-ROUTER destination address for link-level hellos

2. TTL value would be the system default TTL

3. Targeted Hellos SHOULD carry Targeted Hello Optional TLV (Defined in this document.)

4. Holdtime SHOULD NOT be set as 0xffff by a targeted hello sender, and such hellos should be discarded up on receive.

3.3. Address in Hello message

When sending targeted hellos, the sender SHOULD send with its primary reachable address (may be its loopback address) as the source address for the hellos. The other addresses that are relevant SHOULD be added in the secondary address list.

3.4. Timer Values

The timers relevant to this specification are in relation to PIM hello. The recommended timer values are

1: PIM Targeted Hello default refresh time : 60s (2 * Default Link-level hello time)

2: PIM Targeted Hello default hold time : 210s (3.5 times targeted hello default refresh time)

3.5. Targeted Neighbor

A Targeted PIM neighbor is a neighbor-ship established by virtue of exchanging targeted hello messages.

A First-Hop-Router (The initiator) that learns the RP's address would start sending hellos to the known RP address (could be anycast-address).

The RP (The Responder) when it receives this hello, would add sender as a targeted neighbor and would respond to this targeted neighbor from it primary address. The responder SHOULD also include its any-cast address (If available) in the secondary address list. The First-Hop-Router when receiving this hello would form a targeted neighbor with the anycast address.

The RP upon hold-time-out for the neighbor would remove this neighbor and its associated states.

The initiator or responder upon having a need to terminate a targeted neighbor MAY send hello with hold-time as 0.

4. Reliable Connection setup

A reliable connection has to be setup between the First-Hop-Router and RP for reliable registers to happen. Targeted hellos works as the medium for discovery and capability-negotiation between the two peers.

4.1. Active FHR

Once First-Hop-Router and RP discovery each other, First-Hop-Router takes the active role. First-Hop-Router would listen for RP to connect once it forms targeted neighbor-ship with RP. The RP would be expected to use its primary address, which it would have used as the source address in its PIM hellos.

4.2. Connection setup between two RP's

In a network if there happens to be two RP's which are First-Hop-Router's too, then the mechanism could result in two connections getting established. It's desirable to have just one connection instead of two. First-Hop-Router could detect this condition the when it receives hello with targeted hello header identifying that the RP want to be First-Hop-Router too.

In this condition the connection setup could used the procedures stated in PIM over reliable transport [RFC6559]

4.3. Hello Generation ID and reconnect

If RP or First-Hop-Router gets into a situation needing for capability-renegotiation or reconnect, it would change the hello generation ID (gen-ID) to notify it peer to reset all the states and re-init this peering. The trigger for this could be configuration change or local operating parameter change, restart, etc. . .

4.4. Handling Connection or reachability loss

Connection loss or reachability loss could happen for one or more of the following reasons

1: PORT Keep-alive time out

2: Targeted neighbor loss

3: Reliable Connection close

Upon detecting one of these conditions, the connection with the peer SHOULD be closed immediately and the states created by the peer SHOULD be cleared after a grace-period, long enough for the peer to re-establish connection and re-sync the states.

This interval for re-sync would involve the initial time needed for re-establishing the connection, followed by transmission and reception of the states from FHR to RP over the reliable connection.

The ideal interval for this re-sync period could not be predicted, hence the this should be a configurable parameter with default value as 300s.

5. Anycast RP’s

PIM uses Anycast-RP [RFC4610] as a mechanism for RP redundancy. This section describes how anycast-RP would work with this specification.

5.1. Targeted Hellos and Neighbors

An RP that serves an anycast RP address, would know the primary addresses of other RP’s serving the anycast address. These anycast-RP’s would form a full mesh of targeted hello-neighbor-ships. In its targeted hello options tlv, the R bit MUST be set. The secondary address list in the PIM hello message SHOULD include the anycast-addresses that the sender is servicing.

5.2. Anycast-RP connection setup

A full mesh of connection is needed among the anycast-RP’s of the same anycast address. Once targeted neighbor-ship is established, it would use the PIM PORT [RFC6559] procedures to setup reliable connection among them.

5.3. Anycast-RP state sync

An anycast-RP that gets the register state from a peer who’s address is in the RP-set of address for the given group would update the register state and would retain the state. If the peer address is not in the RP-set address for the RP-group range, then the RP would replicate the state to all the other RP’s in the RP-set. This procedure and forwarding rules are similar to the existing forwarding rules in Anycast-rp [RFC4610] register specification.

An RP should identify register state as a combinations of (source, group, ’PORT connection’). Where ’PORT connection’ is the reliable connection with the PORT peer which had reported this s,g. Following considerations are made for a register-state identity.

A. Reconnect: Connection between RP and First-Hop-Router could get re-established for various reasons. The register-states would get retransmitted over the new connection. Then it should be possible for RP to identify and timeout register-states on the old connection and retain the right set of states.

B. DR-change: When DR in the First-Hop-LAN changes, a new First-Hop-Router would be retransmitting the same set of SG’s that are already known and the old DR would be withdrawing the states advertised by it.

C. FHR primary address change: In this case too connection would get re-established and state handling would be similar to case A.

D. Multi-homed sources (but not on same LAN): In this case different First-Hop-Routers could be sending the same register-states. Then RP should be capable of identifying register-state along with the peer.

5.4. Anycast-RP change

In the event of nearest anycast-RP changing over to a different router, First-Hop-Router would detect that when it starts receiving PIM hellos with a different primary address for the same anycast address. This can also happen if the primary address of present anycast-RP has changed.

Upon detecting this scenario, the First-Hop-Router would establish connection and transmit its states to the new peer. Subsequently older connection would get terminated due to neighbor timeout.

5.5. Anycast-RP with MSDP

MSDP [RFC3618] is an alternative mechanism for 'active multicast stream state' synchronization between RP's. When MSDP is used, PIM's anycast synchronization need not be used. An anycast-RP network could use MSDP instead of PIM procedures for state synchronization among anycast-RP's. This document does not state any change in MSDP specification and usage

In such deployments, PIM will not have RP-set configured. As RP-set address is not available PIM procedures for Anycast-RP synchronization does not apply.

MSDP being a soft-state oriented protocol, it depends on frequent state refreshes over the reliable TCP transport. The support for mesh-groups in MSDP could be advantageous in some case.

6. PIM-registers and Interoperation with legacy PIM nodes

It may not be possible for PIM node to migrate altogether onto a PORT-registers in one go. Also there could be a few nodes in the network, which may not support PORT register states. This section states how both could interoperate.

6.1. Initial packet-loss avoidance with PORT

If its found that a few streams in the multicast network has to have initial packets to be delivered to the receiver, the existing PIM register mechanism could be used for them. For these streams a PORT register-stop message would be sent by the RP to First-Hop-Router.

6.2. First-Hop-Router does not support PORT

If the First-Hop-Router is not capable of doing PORT-register, then it would not establish targeted hello neighbor-ship with the RP. Hence reliable connection also would not be established. To handle such scenarios RP should accept PIM register messages and should respond to them with register-stop messages.

6.3. RP does not support PORT

If the RP is not capable of doing PORT-register, then it would not respond to the targeted hellos from the RP. Hence reliable connection also would not be established. In this case First-Hop-Router could sent existing packet registers to RP.

6.4. Data-Register free operations

If initial packet loss is acceptable in a multicast network, then Data-Registers could be avoided altogether in such networks. In such network PORT-Register-state specified in this document alone would be supported.

7. PORT message

This document defines new PORT register state message and PORT register-stop messages, to the existing messages in PORT specification.

7.1. PORT register message TLV

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type = 3 (suggested)      |        Message Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Reserved                   | Exp.  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|B|N|A|                      Reserved-1                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            src addr-1                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            grp addr-1                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
z                            2, 3, . . .                        z
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|B|N|A|                      Reserved-n                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            src addr-n                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            grp addr-n                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: PORT Register State Message for IPv4

Type: This is subject to IANA allocation. Suggested is 3 which is the next unused type.

Length: Length in bytes for the value part of the Type/Length/Value In this case it would be 12 * (number of sg’s present in the register message.) + 4

B: As specified in [RFC4601] (set as 0 on send and ignore when received)

N: As specified in [RFC4601] (set as 1 on send and ignore when received.)

A: This flag signifies if the SG is to be Added or Deleted. When cleared, it indicates that the First-Hop-Router is withdrawing the SG .

src addr-x : This is the 4-byte source address of an ipv4 stream with out any encoding.

grp addr-x : This is the 4-byte group address of an ipv4 stream with out any encoding.

Reserved: Set to zero on transmission and ignored on receipt. These reserved bits are for properties that apply to the entire message.

Reserved-n: Set to zero on transmission and ignored on receipt. These reserved bits are for properties that apply to any particular sg.

Exp: : For experimental use.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |     Type = 5 (suggested)      |        Message Length         |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                            Reserved                   | Exp.  |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |B|N|A|                      Reserved-1                         |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                            src addr-1                         z
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    z                             . . .                             |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                            grp addr-1                         z
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    z                             . . .                             |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    z                            2, 3, . . .                        z
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |B|N|A|                      Reserved-n                         |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                            src addr-n                         z
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    z                             . . .                             |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                            grp addr-n                         z
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    z                             . . .                             |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: PORT Register State Message for IPv6

Type: This is subject to IANA allocation. Suggested is 5 which is the next unused type.

Length: Length in bytes for the value part of the Type/Length/Value In this case it would be 36 * (number of sg’s present in the register message.) + 4

B: As specified in [RFC4601] (set as 0 on send and ignore when received)

N: As specified in [RFC4601] (set as 1 on send and ignore when received.)

A: This flag signifies if the SG is to be Added or Deleted. When cleared, it indicates that the First-Hop-Router is withdrawing the SG .

src addr-x : This is the 16-byte source address of an ipv6 stream with out any encoding.

grp addr-x : This is the 16-byte group address of an ipv6 stream with out any encoding.

Reserved: Set to zero on transmission and ignored on receipt. These reserved bits are for properties that apply to the entire message.

Reserved-n: Set to zero on transmission and ignored on receipt. These reserved bits are for properties that apply to any particular SG.

Exp: : For experimental use.

7.2. Sending and receiving PORT register messages

The First-Hop-Router upon learning a new stream would send a register state add message to the corresponding RP. If the reliable connection got setup later, then once the connection becomes established it would send the entire list of streams active with it.

When KAT timer for a multicast stream expires, it would send an update to RP to remove that stream from its list.

An RP would maintain a database of multicast streams (src, grp) active along with the peer from which it had learned it. If the receiver RP is an anycast RP, it SHOULD re-transmit this register state message to each of the other anycast RP. An RP SHOULD not re-transmit a register state message it received from an another anycast-RP.

7.3. PORT register-stop message TLV

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type = 4 (suggested)      |        Message Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Reserved                   | Exp.  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Reserved-1                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            src addr-1                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            grp addr-1                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
z                          2, 3, . . .                          z
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Reserved-n                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            src addr-n                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            grp addr-n                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: PORT Register Stop Message for IPv4

Type: This is subject to IANA allocation. Suggested is 4 which is the next unused type.

Length: Length in bytes for the value part of the Type/Length/Value In this case it would be 12 * (number of sg’s present in the register message.) + 4

src addr-x : This is the 4-byte source address of an ipv4 stream with out any encoding.

grp addr-x : This is the 4-byte group address of an ipv4 stream with out any encoding.

Reserved: Set to zero on transmission and ignored on receipt. These reserved bits are for properties that apply to the entire message.

Reserved-n: Set to zero on transmission and ignored on receipt. These reserved bits are for properties that apply to any particular sg.

Exp: : For experimental use.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type = 6 (suggested)      |        Message Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Reserved                       | Exp.  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Reserved-1                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          src addr-1                           z
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
z                             . . .                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          grp addr-1                           z
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
z                             . . .                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
z                          2, 3, . . .                          z
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Reserved-n                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          src addr-n                           z
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
z                             . . .                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          grp addr-n                           z
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
z                             . . .                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5: PORT Register Stop Message for IPv6

Type: This is subject to IANA allocation. Suggested is 4 which is the next unused type.

Length: Length in bytes for the value part of the Type/Length/Value In this case it would be 36 * (number of sg’s present in the register message.) + 4

src addr-x : This is the 16-byte source address of an ipv6 stream with out any encoding.

grp addr-x : This is the 16-byte group address of an ipv6 stream with out any encoding.

Reserved: Set to zero on transmission and ignored on receipt. These reserved bits are for properties that apply to the entire message.

Reserved-n: Set to zero on transmission and ignored on receipt. These reserved bits are for properties that apply to any particular sg.

Exp: : For experimental use.

7.4. Sending and receiving PORT register stop messages

PORT register-stop messages are send only as a response to receiving packet registers from a PIM peer, with which a reliable connection has been established. If reliable connection is not available, the RP should consider the peer as a legacy node and should respond to this PIM register-stop message as defined in PIM-SM [RFC4601]

The First-Hop-Router up on receiving PORT-Register-Stop message should treat that as an indication from RP that it does not require the packets over the PIM tunnel and should stop sending register messages.

7.5. PORT Keep-Alive Message

The PORT Keep-alive messages as specified in PIM over Reliable Transport [RFC6559] would be used to check the liveliness of the peer and the reliable session

8. Management Considerations

PORT-register is capable of configuration free operations. But its recommended to have it as configuration controlled.

Implementation should provide knobs needed to stop supporting data-registers on a router.

9. IANA Considerations

This specification introduces new TLV in PIM hello and in PIM PORT messages. Hence the tlv ids for these needs IANA allocation

9.1. PIM Hello Options TLV

The following Hello TLV types needs IANA allocation. Suggested values for these are provided below.

Value	Length	Name	Reference
36 (suggested)	4 (Fixed)	Targeted-Hello-Options	This document

Table 1: Suggested values for PIM Hello TLV type for IANA allocation

9.2. PIM PORT Message Type

The following PIM PORT message TLV types needs IANA allocation. Suggested values for these are provided below.

Value	Name	Reference
3 (suggested)	PORT Register-state for Ipv4	This document
4 (suggested)	PORT Register-stop for Ipv4	This document
5 (suggested)	PORT Register-state for Ipv6	This document
6 (suggested)	PORT Register-stop for Ipv6	This document

Table 2: Suggested values for PIM PORT TLV type for IANA allocation

10. Security Considerations

10.1. PIM Register Threats

PIM register is considered as security vulnerability for PIM networks. [RFC4609] The concern arises mainly due to the existing PIM register protocol design where in any remote node could start sending line-rate multicast traffic as PIM registers due to malfunction, mis-configuration or from a malicious remote node.

10.2. Targeted Hello Threats

This document introduces targeted hellos. This could be a seen as a new security threat. Targeted hellos are part of other IETF protocol implementations, which are widely deployed. In future introduction of a mechanism similar to those stated in RFC 7349 [RFC7349] could be used in PIM.

10.3. TCP or SCTP security threats

The security perception for this is stated in [RFC6559].

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3618] Fenner, B. and D. Meyer, "Multicast Source Discovery Protocol (MSDP)", RFC 3618, October 2003.

[RFC4601] Fenner, B., Handley, M., Holbrook, H., and I. Kouvelas, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", RFC 4601, August 2006.

[RFC4609] Savola, P., Lehtonen, R., and D. Meyer, "Protocol Independent Multicast - Sparse Mode (PIM-SM) Multicast Routing Security Issues and Enhancements", RFC 4609, October 2006.

[RFC4610] Farinacci, D. and Y. Cai, "Anycast-RP Using Protocol Independent Multicast (PIM)", RFC 4610, August 2006.

[RFC6559] Farinacci, D., Wijnands, IJ., Venaas, S., and M. Napierala, "A Reliable Transport Mechanism for PIM", RFC 6559, March 2012.

11.2. Informative References

[RFC7349] Zheng, L., Chen, M., and M. Bhatia, "LDP Hello Cryptographic Authentication", RFC 7349, August 2014.

Authors’ Addresses

Anish Peter (editor)
Juniper Networks, Inc.
Electra, Exora Business Park
Bangalore, KA 560103
India

Email: anishp@juniper.net

Robert Kebler
Juniper Networks, Inc.
1194 N. Mathilda Ave.
Sunnyvale, CA 94089
US

Email: rkebler@juniper.net

Vikram Nagarajan
Juniper Networks, Inc.
Electra, Exora Business Park
Bangalore, KA 560103
India

Email: vikramna@juniper.net

Protocol Independent Multicast (pim)   A. Peter, Ed.
Internet-Draft   Individual contributor
Intended status: Standards Track   R. Kebler
Expires: September 20, 2018   V. Nagarajan
                                                  Juniper Networks, Inc.
                                                               T. Eckert
                            Huawei USA - Futurewei Technologies Inc.
                                                               S. Venaas
                                                     Cisco Systems, Inc.
                                                          March 19, 2018

Reliable Transport For PIM Register States
draft-anish-reliable-pim-registers-02

Abstract

This document introduces a hard-state, reliable transport for the existing PIM-SM registers states. This eliminates the needs for periodic NULL-registers and register-stop in response to each data-register or NULL-registers.

This specification uses the existing PIM reliability mechanisms defined by PIM Over Reliable Transport [RFC6559]. This is simply a means to transmit reliable PIM messages and does not require the support for Join/Prune messages over PORT as defined in [RFC6559].

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC2119 [RFC2119].

Status of This Memo

This Internet-Draft will expire on September 20, 2018.

Copyright Notice

Table of Contents

1. Introduction

Protocol Independent Multicast-Sparse Mode Register mechanism serves the following purposes.

a, With a register, First-Hop-Router (FHR) informs the RP (that way the network) that a particular multicast stream is active

b, A register helps avoid initial packet loss. (Initial packet loss could happen in an anycast-RP deployment even when packet registers are used.)

c, Through its periodic refreshes register keeps RP informed about the aliveness of this multicast stream.

As it is defined in [RFC4601] , register mechanisms face limitations, when the number of multicast streams on the network is high, especially when one RP is expected to serve a large number of streams. These problems are mainly due to these factors.

a, PIM register needs control-plane and data-plane intervention to handle it.

b, Due to the nature of PIM register, First-Hop-Router and RP now needs to maintain states and timers for each register state entry.

c, PIM register's requirements for periodic refresh and expiry, is quite aggressive and makes them vulnerable when the PIM speaker could not find cycles to meet these needs

To take for instance a major multicast application the IPTV. With the streaming servers connected to FHR. A restarting, FHR would result in a burst of register messages at line rate. The RP may get overloaded with packet registers. Which will continue untill RP is able to create states and do a register-stop. In the meantime many flows may go unserviced due to drops. In addition to affecting multicast streams it may lead to starvation for other processing done by the controlplane application. With Anycast RP, this becomes even more tricky due to the control-plane’s job to forward the registers to the rp-set.

In general: PIM registers have limitation in connections across WAN. It has no flow-control mechanisms, making PIM not compatible with IETF transport/congestion control expectations. It is challenging to deployed it over WAN or other bandwidth limited networks. High amount of state: periodic retransmission creates undesirable processing load. Especially with larger mesh-groups (re-send same (S,G) N-times, periodically).

2. Reliable Register Overview

Reliable PIM register extends PIM PORT [RFC6559] to have PIM register states to be sent over a reliable transport.

This document introduces ’targeted’ hellos between any two PIM peers. This helps in capability negotiation and discovery between two PIM speakers (FHR and RP in the context of this document). Once this discovery happens, First-Hop-Router would setup a reliable transport connection based on the negotiated parameters.

Over this reliable connection, First-Hop-Router would start sending to RP the source and group addresses of the multicast streams active with it. When any of this stream stops, First-Hop-Router would sent an update to RP about the streams that have stopped. This way once a reliable connection is setup, First-Hop-Router would update RP with its existing active multicast streams. Subsequently it would sent incremental updates about the change to RP.

For a multicast application that may demand initial packets or for bursty sources existing data-registers may be used. For them the RP would now respond with a ’reliable’-register-stop, which could persist until the First-Hop-Router withdraws the register-state.

3. Targeted Hellos

PIM hellos defined in PIM-SM [RFC4601] confines them to link level. This document extends these hellos to support ’targeted’ hellos.

Targeted hellos are identical to existing hellos messages except that they would have an unicast address as its destination address. It would traverse multiple hops using the unicast routing to reach the targeted hello neighbor.

3.1. New Hello Optional TLV’s

Option Type: Targeted hello

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type = H1 (for alloc)     |           Length = 4          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|F|R|                        Reserved                  |  Exp  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: PIM Hello Optional TLV

Assigned Hello Type values can be found in IANA PIM registry.

Type: This is subject to IANA allocation. Its stated as H1 for reference

Length: Length in bytes for the value part of the Type/Length/Value encoding fixed as 4.

F: To be set by a router that wants to be a First-Hop-Router.

R: To be set by a RP that is capable taking the role of an RP as per the current states.

Reserved: Set to zero on transmission and ignored on receipt.

Exp: For experimental use [RFC3692]. One expected use of these bits would be to signal experimental capabilities. For example, if a router supports an experimental feature, it may set a bit to indicate this. The default behavior, unless a router supports a particular experiment, is to keep these bits reset and ignore the bits on receipt.

3.2. Differences from Link-Level hellos

The Major differences that Link-Level-Hellos have over Interface hellos are,

1. Destination address would be an unicast address unlike ALL-PIM-ROUTER destination address for link-level hellos

2. TTL value would be the system default TTL

3. Targeted Hellos SHOULD carry Targeted Hello Optional TLV (Defined in this document.)

4. Holdtime SHOULD NOT be set as 0xffff by a targeted hello sender, and such hellos should be discarded up on receive.

3.3. Address in Hello message

When sending targeted hellos, the sender SHOULD send with its primary reachable address (may be its loopback address) as the source address for the hellos. The other addresses that are relevant SHOULD be added in the secondary address list.

3.4. Timer Values

The timers relevant to this specification are in relation to PIM hello. The recommended timer values are

1: PIM Targeted Hello default refresh time : 60s (2 * Default Link-level hello time)

2: PIM Targeted Hello default hold time : 210s (3.5 times targeted hello default refresh time)

3.5. Targeted Neighbor

A Targeted PIM neighbor is a neighbor-ship established by virtue of exchanging targeted hello messages.

A First-Hop-Router (The initiator) that learns the RP’s address would start sending hellos to the known RP address (could be anycast-address).

The RP (The Responder) when it receives this hello, would add sender as a targeted neighbor and would respond to this targeted neighbor from it primary address. The responder SHOULD also include its any-cast address (If available) in the secondary address list. The First-Hop-Router when receiving this hello would form a targeted neighbor with the anycast address.

The RP upon hold-time-out for the neighbor would remove this neighbor and its associated states.

The initiator or responder upon having a need to terminate a targeted neighbor MAY send hello with hold-time as 0.

4. Reliable Connection setup

A reliable connection has to be setup between the First-Hop-Router and RP for reliable registers to happen. Targeted hellos works as the medium for discovery and capability-negotiation between the two peers.

4.1. Active FHR

Once First-Hop-Router and RP discovery each other, First-Hop-Router takes the active role. First-Hop-Router would listen for RP to connect once it forms targeted neighbor-ship with RP. The RP would be expected to use its primary address, which it would have used as the source address in its PIM hellos.

4.2. Connection setup between two RP's

In a network if there happens to be two RP's which are First-Hop-Router's too, then the mechanism could result in two connections getting established. It's desirable to have just one connection instead of two. First-Hop-Router could detect this condition the when it receives hello with targeted hello header identifying that the RP want to be First-Hop-Router too.

In this condition the connection setup could used the procedures stated in PIM over reliable transport [RFC6559]

4.3. Hello Generation ID and reconnect

If RP or First-Hop-Router gets into a situation needing for capability-renegotiation or reconnect, it would change the hello generation ID (gen-ID) to notify it peer to reset all the states and re-init this peering. The trigger for this could be configuration change or local operating parameter change, restart, etc. . .

4.4. Handling Connection or reachability loss

Connection loss or reachability loss could happen for one or more of the following reasons

1: PORT Keep-alive time out

2: Targeted neighbor loss

3: Reliable Connection close

Upon detecting one of these conditions, the connection with the peer SHOULD be closed immediately and the states created by the peer

SHOULD be cleared after a grace-period, long enough for the peer to re-establish connection and re-sync the states.

This interval for re-sync would involve the initial time needed for re-establishing the connection, followed by transmission and reception of the states from FHR to RP over the reliable connection.

The ideal interval for this re-sync period could not be predicted, hence the this should be a configurable parameter with default value as 300s.

5. Anycast RP’s

PIM uses Anycast-RP [RFC4610] as a mechanism for RP redundancy. This section describes how anycast-RP would work with this specification.

5.1. Targeted Hellos and Neighbors

An RP that serves an anycast RP address, would know the primary addresses of other RP’s serving the anycast address. These anycast-RP’s would form a full mesh of targeted hello-neighbor-ships. In its targeted hello options tlv, the R bit MUST be set. The secondary address list in the PIM hello message SHOULD include the anycast-addresses that the sender is servicing.

5.2. Anycast-RP connection setup

A full mesh of connection is needed among the anycast-RP’s of the same anycast address. Once targeted neighbor-ship is established, it would use the PIM PORT [RFC6559] procedures to setup reliable connection among them.

5.3. Anycast-RP state sync

An anycast-RP that gets the register state from a peer who’s address is in the RP-set of address for the given group would update the register state and would retain the state. If the peer address is not in the RP-set address for the RP-group range, then the RP would replicate the state to all the other RP’s in the RP-set. This procedure and forwarding rules are similar to the existing forwarding rules in Anycast-rp [RFC4610] register specification.

An RP should identify register state as a combinations of (source, group, ’PORT connection’). Where ’PORT connection’ is the reliable connection with the PORT peer which had reported this s,g. Following considerations are made for a register-state identity.

A. Reconnect: Connection between RP and First-Hop-Router could get re-established for various reasons. The register-states would get retransmitted over the new connection. Then it should be possible for RP to identify and timeout register-states on the old connection and retain the right set of states.

B. DR-change: When DR in the First-Hop-LAN changes, a new First-Hop-Router would be retransmitting the same set of SG's that are already known and the old DR would be withdrawing the states advertised by it.

C. FHR primary address change: In this case too connection would get re-established and state handling would be similar to case A.

D. Multi-homed sources (but not on same LAN): In this case different First-Hop-Routers could be sending the same register-states. Then RP should be capable of identifying register-state along with the peer.

5.4. Anycast-RP change

In the event of nearest anycast-RP changing over to a different router, First-Hop-Router would detect that when it starts receiving PIM hellos with a different primary address for the same anycast address. This can also happen if the primary address of present anycast-RP has changed.

Upon detecting this scenario, the First-Hop-Router would establish connection and transmit its states to the new peer. Subsequently older connection would get terminated due to neighbor timeout.

5.5. Anycast-RP with MSDP

MSDP [RFC3618] is an alternative mechanism for 'active multicast stream state' synchronization between RP's. When MSDP is used, PIM's anycast synchronization need not be used. An anycast-RP network could use MSDP instead of PIM procedures for state synchronization among anycast-RP's. This document does not state any change in MSDP specification and usage

In such deployments, PIM will not have RP-set configured. As RP-set address is not available PIM procedures for Anycast-RP synchronization does not apply.

MSDP being a soft-state oriented protocol, it depends on frequent state refreshes over the reliable TCP transport. The support for mesh-groups in MSDP could be advantageous in some case.

6. PIM-registers and Interoperation with legacy PIM nodes

It may not be possible for PIM node to migrate altogether onto a PORT-registers in one go. Also there could be a few nodes in the network, which may not support PORT register states. This section states how both could interoperate.

6.1. Initial packet-loss avoidance with PORT

If its found that a few streams in the multicast network has to have initial packets to be delivered to the receiver, the existing PIM register mechanism could be used for them. For these streams a PORT register-stop message would be sent by the RP to First-Hop-Router.

6.2. First-Hop-Router does not support PORT

If the First-Hop-Router is not capable of doing PORT-register, then it would not establish targeted hello neighbor-ship with the RP. Hence reliable connection also would not be established. To handle such scenarios RP should accept PIM register messages and should respond to them with register-stop messages.

6.3. RP does not support PORT

If the RP is not capable of doing PORT-register, then it would not respond to the targeted hellos from the RP. Hence reliable connection also would not be established. In this case First-Hop-Router could sent existing packet registers to RP.

6.4. Data-Register free operations

If initial packet loss is acceptable in a multicast network, then Data-Registers could be avoided altogether in such networks. In such network PORT-Register-state specified in this document alone would be supported.

7. PORT message

This document defines new PORT register state message and PORT register-stop messages, to the existing messages in PORT specification.

7.1. PORT register message TLV

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type = P1 (for alloc)     |        Message Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Reserved                   | Exp.  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|B|N|A|                      Reserved-1                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            src addr-1                         z
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
z                            grp addr-1                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
z                            2, 3, . . .                        z
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|B|N|A|                      Reserved-n                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            src addr-n                         z
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
z                            grp addr-n                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: PORT Register State Message

Type: This is subject to IANA allocation. It would be next unallocated value, which is referred untill allocation as P1.

Length: Length in bytes for the value part of the Type/Length/Value

B: As specified in [RFC4601] (set as 0 on send and ignore when received)

N: As specified in [RFC4601] (set as 1 on send and ignore when received.)

A: This flag signifies if the SG is to be Added or Deleted. When cleared, it indicates that the First-Hop-Router is withdrawing the SG .

src addr-x : This is the encoded source address of an ipv4/ipv6 stream

grp addr-x : This is the encoded group address of an ipv4/ipv6 stream

Reserved: Set to zero on transmission and ignored on receipt. These reserved bits are for properties that apply to the entire message.

Reserved-n: Set to zero on transmission and ignored on receipt. These reserved bits are for properties that apply to any particular sg.

Exp: : For experimental use.

7.2. Sending and receiving PORT register messages

The First-Hop-Router upon learning a new stream would send a register state add message to the corresponding RP. If the reliable connection got setup later, then once the connection becomes established it would send the entire list of streams active with it.

When KAT timer for a multicast stream expires, it would send an update to RP to remove that stream from its list.

An RP would maintain a database of multicast streams (src, grp) active along with the peer from which it had learned it. If the receiver RP is an anycast RP, it SHOULD re-transmit this register state message to each of the other anycast RP. An RP SHOULD not re-transmit a register state message it received from an another anycast-RP.

7.3. PORT register-stop message TLV

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type = P2(for alloc)      |         Message Length        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Reserved                | Exp.     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Reserved-1                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            src addr-1                         z
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
z                            grp addr-1                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
z                           2, 3, . . .                         z
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Reserved-n                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            src addr-n                         z
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
z                            grp addr-n                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: PORT Register Stop Message

Type: This is subject to IANA allocation. It would be next unallocated value, which is referred untill allocation as P2.

Length: Length in bytes for the value part of the Type/Length/Value

src addr-x : This is the encoded source address of an ipv4/ipv6 stream

grp addr-x : This is the encoded group address of an ipv4/ipv6 stream

Reserved: Set to zero on transmission and ignored on receipt. These reserved bits are for properties that apply to the entire message.

Reserved-n: Set to zero on transmission and ignored on receipt. These reserved bits are for properties that apply to any particular sg.

Exp: : For experimental use.

7.4. Sending and receiving PORT register stop messages

PORT register-stop messages are send only as a response to receiving packet registers from a PIM peer, with which a reliable connection has been established. If reliable connection is not available, the RP should consider the peer as a legacy node and should respond to this PIM register-stop message as defined in PIM-SM [RFC4601]

The First-Hop-Router up on receiving PORT-Register-Stop message should treat that as an indication from RP that it does not require the packets over the PIM tunnel and should stop sending register messages.

7.5. PORT Keep-Alive Message

The PORT Keep-alive messages as specified in PIM over Reliable Transport [RFC6559] would be used to check the liveliness of the peer and the reliable session

8. Management Considerations

PORT-register is capable of configuration free operations. But its recommended to have it as configuration controlled.

Implementation should provide knobs needed to stop supporting data-registers on a router.

9. IANA Considerations

This specification introduces new TLV in PIM hello and in PIM PORT messages. Hence the tlv ids for these needs IANA allocation

9.1. PIM Hello Options TLV

The following Hello TLV types needs IANA allocation. Place holder are kept to differentiate the different types.

Value	Length	Name	Reference
H1 (next-available)	4 (Fixed)	Targeted-Hello-Options	This document

Table 1: Place holder values for PIM Hello TLV type until IANA allocation

9.2. PIM PORT Message Type

The following PIM PORT message TLV types needs IANA allocation. Place holder are kept to differentiate the different types.

Value	Name	Reference
P1 (Next available)	PORT Register-state	This document
P2 (Next available)	PORT Register-stop	This document

Table 2: Place holder values for PIM PORT TLV type for IANA allocation

10. Security Considerations

10.1. PIM Register Threats

PIM register is considered as security vulnerability for PIM networks. [RFC4609] The concern arises mainly due to the existing PIM register protocol design where in any remote node could start sending line-rate multicast traffic as PIM registers due to malfunction, mis-configuration or from a malicious remote node.

10.2. Targeted Hello Threats

This document introduces targeted hellos. This could be a seen as a new security threat. Targeted hellos are part of other IETF protocol implementations, which are widely deployed. In future introduction of a mechanism similar to those stated in RFC 7349 [RFC7349] could be used in PIM.

10.3. TCP or SCTP security threats

The security perception for this is stated in [RFC6559].

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3618] Fenner, B., Ed. and D. Meyer, Ed., "Multicast Source Discovery Protocol (MSDP)", RFC 3618, DOI 10.17487/RFC3618, October 2003, <https://www.rfc-editor.org/info/rfc3618>.

[RFC4601] Fenner, B., Handley, M., Holbrook, H., and I. Kouvelas, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", RFC 4601, DOI 10.17487/RFC4601, August 2006, <https://www.rfc-editor.org/info/rfc4601>.

[RFC4609] Savola, P., Lehtonen, R., and D. Meyer, "Protocol Independent Multicast - Sparse Mode (PIM-SM) Multicast Routing Security Issues and Enhancements", RFC 4609, DOI 10.17487/RFC4609, October 2006, <https://www.rfc-editor.org/info/rfc4609>.

[RFC4610] Farinacci, D. and Y. Cai, "Anycast-RP Using Protocol Independent Multicast (PIM)", RFC 4610, DOI 10.17487/RFC4610, August 2006, <https://www.rfc-editor.org/info/rfc4610>.

[RFC6559] Farinacci, D., Wijnands, IJ., Venaas, S., and M. Napierala, "A Reliable Transport Mechanism for PIM", RFC 6559, DOI 10.17487/RFC6559, March 2012, <https://www.rfc-editor.org/info/rfc6559>.

11.2. Informative References

[RFC7349] Zheng, L., Chen, M., and M. Bhatia, "LDP Hello Cryptographic Authentication", RFC 7349, DOI 10.17487/RFC7349, August 2014, <https://www.rfc-editor.org/info/rfc7349>.

Authors' Addresses

Anish Peter (editor)
Individual contributor
Brunton Road
Bangalore, KA 560001
India

Email: anish.ietf@gmail.com

Robert Kebler
Juniper Networks, Inc.
1194 N. Mathilda Ave.
Sunnyvale, CA 94089
US

Email: rkebler@juniper.net

Vikram Nagarajan
Juniper Networks, Inc.
Electra, Exora Business Park
Bangalore, KA 560103
India

Email: vikramna@juniper.net

Toerless Eckert
Huawei USA - Futurewei Technologies Inc.

Email: tte+ietf@cs.fau.de

Stig Venaas
Cisco Systems, Inc.

Email: stig@cisco.com

PIM Working Group H. Zhao
Internet Draft Ericsson
Intended status: Standards Track X. Liu
Expires: August 23, 2018 Jabil
Y. Liu
Huawei
M. Sivakumar
Cisco
A. Peter
Individual

February 24, 2018

A Yang Data Model for IGMP and MLD Snooping
draft-ietf-pim-igmp-mld-snooping-yang-01.txt

Abstract

This document defines a YANG data model that can be used to configure and manage Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD) Snooping devices.

Copyright Notice

Table of Contents

1. Introduction

This document defines a YANG [RFC6020] data model for the management of Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD) Snooping devices.

This data model follows the Guidelines for YANG Module Authors NMDA)[draft-dsdt-nmda-guidelines-01]. The "Network Management Datastore Architecture" (NMDA) adds the ability to inspect the current operational values for configuration, allowing clients to use identical paths for retrieving the configured values and the operational values.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119].

The terminology for describing YANG data models is found in[RFC6020].

1.2. Tree Diagrams

A simplified graphical representation of the data model is used in this document. The meaning of the symbols in these diagrams is as follows:

- Brackets "[" and "]" enclose list keys.

- Abbreviations before data node names: "rw" means configuration (read-write), and "ro" means state data (read-only).

- Symbols after data node names: "?" means an optional node, "!" means a presence container, and "*" denotes a list and leaf-list.

- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").

- Ellipsis ("...") stands for contents of subtrees that are not shown.

2. Design of Data Model

The model covers Considerations for Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD) Snooping Switches [RFC4541].

The goal of this document is to define a data model that provides a common user interface to IGMP and MLD Snooping. There is very information that is designated as "mandatory", providing freedom for vendors to adapt this data model to their respective product implementations.

2.1. Overview

The IGMP and MLD Snooping YANG module defined in this document has all the common building blocks for the IGMP and MLD Snooping protocol.

The YANG module includes IGMP and MLD Snooping instances definition, instance references in the scenario of BRIDGE, VPLS. The module also includes the RPC methods for clearing the specified IGMP and MLD Snooping.

This YANG model follows the Guidelines for YANG Module Authors (NMDA) [draft-dsdt-nmda-guidelines-01]. This NMDA ("Network Management Datastore Architecture") architecture provides an architectural framework for datastores as they are used by network management protocols such as NETCONF [RFC6241], RESTCONF [RFC8040] and the YANG [RFC7950] data modeling language..

2.2. IGMP and MLD Snooping Instances

The YANG module defines IGMP and MLD Snooping instance. The instance will be referenced in all kinds of scenarios to configure IGMP and MLD Snooping. The attribute who could be read and written shows configuration data. The read-only attribute shows state data. The key attribute is name.

```
module: ietf-igmp-mld-snooping

    +--rw igmp-snooping-instances

    |  +--rw igmp-snooping-instance* [name]

    |     +--rw name                              string

    |     +--rw type?                             enumeration

    |     +--rw enable?                           boolean {admin-enable}?

    |     +--rw forwarding-mode?                  enumeration

    |     +--rw explicit-tracking?                boolean {explicit-tracki
ng}?

    |     +--rw exclude-lite?                     boolean {exclude-lite}?

    |     +--rw send-query?                       boolean

    |     +--rw immediate-leave?                  empty {immediate-leave}?
```

```
       |     +--rw last-member-query-interval?          uint16

       |     +--rw query-interval?                      uint16

       |     +--rw query-max-response-time?             uint16

       |     +--rw require-router-alert?                boolean {require-router-
   alert}?

       |     +--rw robustness-variable?                 uint8

       |     +--rw version?                             uint8

       |     +--rw static-bridge-mrouter-interface*     if:interface-ref {static
-
   mrouter-interface}?

       |     +--rw static-l2vpn-mrouter-interface-ac*   if:interface-ref {static
-
   mrouter-interface}?

       |     +--rw static-l2vpn-mrouter-interface-pw*   l2vpn-instance-pw-ref
   {static-mrouter-interface}?

       |     +--rw querier-source?                      inet:ipv4-address

       |     +--rw static-l2-multicast-group* [group source-addr] {static-l2-
   multicast-group}?

       |     |  +--rw group                        inet:ipv4-address

       |     |  +--rw source-addr                  source-ipv4-addr-type

       |     |  +--rw bridge-outgoing-interface*   if:interface-ref

       |     |  +--rw l2vpn-outgoing-ac*           l2vpn-instance-ac-ref

       |     |  +--rw l2vpn-outgoing-pw*           l2vpn-instance-pw-ref

       |     +--ro entries-count?                       uint32

       |     +--ro bridge-mrouter-interface*            if:interface-ref

       |     +--ro l2vpn-mrouter-interface-ac*          if:interface-ref

       |     +--ro l2vpn-mrouter-interface-pw*          l2vpn-instance-pw-ref

       |     +--ro group* [address]

       |        +--ro address          inet:ipv4-address

       |        +--ro mac-address?     yang:phys-address
```

```
       |        +--ro expire?          uint32
       |        +--ro up-time?         uint32
       |        +--ro last-reporter?   inet:ipv4-address
       |        +--ro source* [address]
       |           +--ro address                        inet:ipv4-address
       |           +--ro bridge-outgoing-interface*   if:interface-ref
       |           +--ro l2vpn-outgoing-ac*           l2vpn-instance-ac-ref
       |           +--ro l2vpn-outgoing-pw*           l2vpn-instance-pw-ref
       |           +--ro up-time?                     uint32
       |           +--ro expire?                      uint32
       |           +--ro host-count?                  uint32 {explicit-tracking}
?
       |           +--ro last-reporter?               inet:ipv4-address
       |           +--ro host* [host-address] {explicit-tracking}?
       |              +--ro host-address        inet:ipv4-address
       |              +--ro host-filter-mode?   enumeration
       +--rw mld-snooping-instances
       |  +--rw mld-snooping-instance* [name]
       |     +--rw name                                  string
       |     +--rw type?                                 enumeration
       |     +--rw enable?                               boolean {admin-enable}?
       |     +--rw forwarding-mode?                      enumeration
       |     +--rw explicit-tracking?                    boolean {explicit-tracki
ng}?
       |     +--rw exclude-lite?                         boolean {exclude-lite}?
       |     +--rw send-query?                           boolean
       |     +--rw immediate-leave?                      empty {immediate-leave}?
```

```
      |     +--rw last-member-query-interval?          uint16

      |     +--rw query-interval?                      uint16

      |     +--rw query-max-response-time?             uint16

      |     +--rw require-router-alert?                boolean {require-router-
   alert}?

      |     +--rw robustness-variable?                 uint8

      |     +--rw version?                             uint8

      |     +--rw static-bridge-mrouter-interface*     if:interface-ref {static
-
   mrouter-interface}?

      |     +--rw static-l2vpn-mrouter-interface-ac*   if:interface-ref {static
-
   mrouter-interface}?

      |     +--rw static-l2vpn-mrouter-interface-pw*   l2vpn-instance-pw-ref
   {static-mrouter-interface}?

      |     +--rw querier-source?                      inet:ipv6-address

      |     +--rw static-l2-multicast-group* [group source-addr] {static-l2-
   multicast-group}?

      |     |  +--rw group                        inet:ipv6-address

      |     |  +--rw source-addr                  source-ipv6-addr-type

      |     |  +--rw bridge-outgoing-interface*   if:interface-ref

      |     |  +--rw l2vpn-outgoing-ac*           l2vpn-instance-ac-ref

      |     |  +--rw l2vpn-outgoing-pw*           l2vpn-instance-pw-ref

      |     +--ro entries-count?                       uint32

      |     +--ro bridge-mrouter-interface*            if:interface-ref

      |     +--ro l2vpn-mrouter-interface-ac*          if:interface-ref

      |     +--ro l2vpn-mrouter-interface-pw*          l2vpn-instance-pw-ref

      |     +--ro group* [address]

      |        +--ro address          inet:ipv6-address

      |        +--ro mac-address?     yang:phys-address
```

```
       |        +--ro expire?          uint32
       |        +--ro up-time?         uint32
       |        +--ro last-reporter?   inet:ipv6-address
       |        +--ro source* [address]
       |           +--ro address                        inet:ipv6-address
       |           +--ro bridge-outgoing-interface*   if:interface-ref
       |           +--ro l2vpn-outgoing-ac*           l2vpn-instance-ac-ref
       |           +--ro l2vpn-outgoing-pw*           l2vpn-instance-pw-ref
       |           +--ro up-time?                      uint32
       |           +--ro expire?                       uint32
       |           +--ro host-count?                   uint32 {explicit-tracking}
?
       |           +--ro last-reporter?                inet:ipv6-address
       |           +--ro host* [host-address] {explicit-tracking}?
       |              +--ro host-address        inet:ipv6-address
      |              +--ro host-filter-mode?   Enumeration
```

2.3. IGMP and MLD Snooping References

The IGMP and MLD Snooping instance could be referenced in the scenario of bridge, L2VPN to configure the IGMP and MLD Snooping. The name of the instance is the key attribute.

```
+--rw bridges
|  +--rw bridge* [name]
|     +--rw name                      name-type
|     +--rw igmp-snooping-instance?   igmp-snooping-instance-ref
|     +--rw mld-snooping-instance?    mld-snooping-instance-ref
|     +--rw component* [name]
|        +--rw name           string
|        +--rw bridge-vlan
|           +--rw vlan* [vid]
|              +--rw vid                      vlan-index-type
|              +--rw igmp-snooping-instance?   igmp-snooping-instance-ref
|              +--rw mld-snooping-instance?    mld-snooping-instance-ref
+--rw l2vpn-instances
   +--rw l2vpn-instance* [name]
      +--rw name                      string
      +--rw igmp-snooping-instance?   igmp-snooping-instance-ref
      +--rw mld-snooping-instance?    mld-snooping-instance-ref
```

2.4. Augment /if:interfaces/if:interface

Augment /if:interfaces/if:interface then add the IGMP MLD SNOOPING related attributes under it. It includes enable, version, static-mrouter-interface, etc.

```
augment /if:interfaces/if:interface:
    +--rw igmp-mld-snooping
       +--rw enable?                      boolean {admin-enable}?
       +--rw version?                     uint8
       +--rw type?                        enumeration
       +--rw static-mrouter-interface
       |  +--rw (static-mrouter-interface)?
       |     +--:(bridge)
       |     |  +--rw bridge-name?           string
       |     |  +--rw vlan-id*               uint32
       |     +--:(l2vpn)
       |        +--rw l2vpn-instance-name?   string
       +--rw static-l2-multicast-group
       |  +--rw (static-l2-multicast-group)?
       |     +--:(bridge)
       |     |  +--rw bridgename?        string
       |     |  +--rw bridge-group-v4* [address]
       |     |  |  +--rw address    inet:ipv4-address
       |     |  |  +--rw source*    inet:ipv4-address
       |     |  |  +--rw vlan-id*   uint32
       |     |  +--rw bridge-group-v6* [address]
```

```
       |     |     +--rw address    inet:ipv6-address
       |     |     +--rw source*    inet:ipv6-address
       |     |     +--rw vlan-id*   uint32
       |     +--:(l2vpn)
       |        +--rw l2vpn-group-v4* [address]
       |        |  +--rw address               inet:ipv4-address
       |        |  +--rw source*               inet:ipv4-address
       |        |  +--rw l2vpn-instance-name?   string
       |        +--rw l2vpn-group-v6* [address]
       |           +--rw address               inet:ipv6-address
       |           +--rw source*               inet:ipv6-address
       |           +--rw l2vpn-instance-name?   string
       +--ro statistics
          +--ro received
          |  +--ro query?                  yang:counter64
          |  +--ro membership-report-v1?   yang:counter64
          |  +--ro membership-report-v2?   yang:counter64
          |  +--ro membership-report-v3?   yang:counter64
          |  +--ro leave?                  yang:counter64
          |  +--ro non-member-leave?       yang:counter64
          |  +--ro pim?                    yang:counter64
          +--ro sent
             +--ro query?                  yang:counter64
             +--ro membership-report-v1?   yang:counter64
             +--ro membership-report-v2?   yang:counter64
```

```
             +--ro membership-report-v3?   yang:counter64

             +--ro leave?                  yang:counter64

             +--ro non-member-leave?       yang:counter64

             +--ro pim?                    yang:counter64
```

2.5. IGMP and MLD Snooping RPC

IGMP and MLD Snooping RPC clears the specified IGMP and MLD Snooping group tables.

```
   rpcs:
       +---x clear-igmp-snooping-groups {rpc-clear-groups}?
       |  +---w input
       |     +---w name?     string
       |     +---w group?    inet:ipv4-address
       |     +---w source?   inet:ipv4-address
       +---x clear-mld-snooping-groups {rpc-clear-groups}?
          +---w input
             +---w name?     string
             +---w group?    inet:ipv6-address
             +---w source?   inet:ipv6-address
```

3. IGMP and MLD Snooping YANG Module

```
<CODE BEGINS> file "ietf-igmp-mld-snooping@2018-02-26.yang"
module ietf-igmp-mld-snooping {
      namespace "urn:ietf:params:xml:ns:yang:ietf-igmp-mld-snooping";
      // replace with IANA namespace when assigned
      prefix ims;

      import ietf-inet-types {
        prefix "inet";
      }

      import ietf-yang-types {
        prefix "yang";
      }

      import ietf-interfaces {
        prefix "if";
      }

     import ietf-l2vpn {
      prefix "l2vpn";
     }

     import ietf-network-instance {
        prefix "ni";
      }


      organization
        "IETF PIM Working Group";

      contact
        "WG Web:   <http://tools.ietf.org/wg/pim/>
         WG List:  <mailto:pim@ietf.org>

         WG Chair: Stig Venaas
                   <mailto:stig@venaas.com>

         WG Chair: Mike McBride
                   <mailto:Michael.McBride@huawei.com>

         Editors:  Hongji Zhao
               <mailto:hongji.zhao@ericsson.com>

               Xufeng Liu
               <mailto:Xufeng_Liu@jabil.com>

               Yisong Liu
               <mailto:liuyisong@huawei.com>
```

```
               Anish Peter
               <mailto:anish.ietf@gmail.com>

               Mahesh Sivakumar
               <mailto:masivaku@cisco.com>

               ";

      description
        "The module defines a collection of YANG definitions common for
        IGMP and MLD Snooping.";

     revision 2018-02-26 {
      description
         "augment /if:interfaces/if:interface";
      reference
          "RFC XXXX: A YANG Data Model for IGMP and MLD Snooping";
     }

     revision 2017-10-24 {
        description
          "Change model definition to fit NMDA standard.";
        reference
          "RFC XXXX: A YANG Data Model for IGMP and MLD Snooping";
      }

     revision 2017-08-14 {
        description
          "using profile to cooperate with ieee-dot1Q-bridge module";
        reference
          "RFC XXXX: A YANG Data Model for IGMP and MLD Snooping";
      }

     revision 2017-06-28 {
        description
          "augment /rt:routing/rt:control-plane-protocols
         augment /rt:routing-state/rt:control-plane-protocols";
        reference
          "RFC XXXX: A YANG Data Model for IGMP and MLD Snooping";
      }

      revision 2017-02-05 {
        description
          "Initial revision.";
        reference
          "RFC XXXX: A YANG Data Model for IGMP and MLD Snooping";
      }

      /*
```

```
       * Features
       */

     feature admin-enable {
        description
          "Support configuration to enable or disable IGMP and MLD
Snooping.";
      }

      feature immediate-leave {
        description
          "Support configuration of immediate-leave.";
      }

      feature join-group {
        description
          "Support configuration of join-group.";
      }

      feature require-router-alert {
        description
          "Support configuration of require-router-alert.";
      }

      feature static-l2-multicast-group {
        description
          "Support configuration of L2 multicast static-group.";
      }

     feature static-mrouter-interface {
        description
          "Support configuration of mrouter interface.";
      }

      feature per-instance-config {
        description
          "Support configuration of each VLAN or l2vpn instance or EVPN
instance.";
      }

     feature rpc-clear-groups {
        description
          "Support to clear statistics by RPC for IGMP and MLD
Snooping.";
      }

     feature explicit-tracking {
        description
          "Support configuration of per instance explicit-tracking
hosts.";
```

```
      }

      feature exclude-lite {
        description
          "Support configuration of per instance exclude-lite.";
      }

      /*
       * Typedefs
       */
    typedef name-type {
      type string {
        length "0..32";
      }
      description
        "A text string of up to 32 characters, of locally determined
        significance.";
    }
    typedef vlan-index-type {
      type uint32 {
        range "1..4094 | 4096..4294967295";
      }
      description
        "A value used to index per-VLAN tables. Values of 0 and 4095
        are not permitted. The range of valid VLAN indices. If the
        value is greater than 4095, then it represents a VLAN with
        scope local to the particular agent, i.e., one without a
        global VLAN-ID assigned to it. Such VLANs are outside the
        scope of IEEE 802.1Q, but it is convenient to be able to
        manage them in the same way using this YANG module.";
      reference
        "IEEE Std 802.1Q-2014: Virtual Bridged Local Area Networks.";
    }

     typedef igmp-snooping-instance-ref {
       type leafref {
         path "/igmp-snooping-instances/igmp-snooping-instance/name";
       }
       description
         "This type is used by data models that need to reference igmp
snooping instance.";
     }

    typedef mld-snooping-instance-ref {
       type leafref {
         path "/mld-snooping-instances/mld-snooping-instance/name";
       }
       description
         "This type is used by data models that need to reference mld
snooping instance.";
```

```
     }

    typedef l2vpn-instance-ac-ref {
      type leafref {
            path "/ni:network-instances/ni:network-instance/l2vpn:endpoint/l2vpn
:name";

      }
      description "l2vpn-instance-ac-ref";
    }

    typedef l2vpn-instance-pw-ref {
      type leafref {
         path "/ni:network-instances/ni:network-instance/l2vpn:endpoint/l2vpn:na
me";
      }
      description "l2vpn-instance-pw-ref";
    }



     typedef source-ipv4-addr-type {
        type union {
          type enumeration {
            enum '*' {
              description
              "Any source address.";
            }
          }
          type inet:ipv4-address;
        }
        description
          "Multicast source IP address type.";
      } // source-ipv4-addr-type

      typedef source-ipv6-addr-type {
        type union {
          type enumeration {
            enum '*' {
              description
              "Any source address.";
            }
          }
          type inet:ipv6-address;
        }
        description
          "Multicast source IP address type.";
      } // source-ipv6-addr-type
```

```
      /*
       * Identities
       */

      /*
       * Groupings
       */


      grouping general-state-attributes {

      description "General State attributes";

        container received {
        config false;
          description "Statistics of received IGMP and MLD Snooping
related packets.";
          uses general-statistics-sent-received;
        }
        container sent {
        config false;
          description "Statistics of sent IGMP and MLD Snooping related
packets.";
          uses general-statistics-sent-received;
        }


      } // general-state-attributes



      grouping instance-config-attributes-igmp-snooping {
        description "IGMP snooping configuration for each VLAN or l2vpn
instance or EVPN instance.";

        uses instance-config-attributes-igmp-mld-snooping;

      leaf querier-source {
          type inet:ipv4-address;
          description "Use the IGMP snooping querier to support IGMP
snooping in a VLAN where PIM and IGMP are not configured.
                  The IP address is used as the source address in
messages.";
        }

       list static-l2-multicast-group {
         if-feature static-l2-multicast-group;
         key "group source-addr";
             description
               "A static multicast route, (*,G) or (S,G).";
```

```
         leaf group {
            type inet:ipv4-address;
            description
            "Multicast group IP address";
         }

         leaf source-addr {
               type source-ipv4-addr-type;
               description
                 "Multicast source IP address.";
             }

         leaf-list bridge-outgoing-interface {
            when "../../type = 'bridge'";
            type if:interface-ref;
            description "Outgoing interface in bridge fowarding";

         }

         leaf-list l2vpn-outgoing-ac {
            when "../../type = 'l2vpn'";
            type l2vpn-instance-ac-ref;
            description "Outgoing ac in l2vpn fowarding";

         }

         leaf-list l2vpn-outgoing-pw {
            when "../../type = 'l2vpn'";
            type l2vpn-instance-pw-ref;
            description "Outgoing pw in l2vpn fowarding";

         }


       } // static-l2-multicast-group

      } // instance-config-attributes-igmp-snooping

      grouping instance-config-attributes-igmp-mld-snooping {
        description
          "IGMP and MLD Snooping configuration of each VLAN.";

        leaf enable {
          if-feature admin-enable;
          type boolean;
          description
            "Set the value to true to enable IGMP and MLD Snooping in
the VLAN instance.";
        }
```

```
      leaf forwarding-mode {
         type enumeration {
            enum "mac" {
             description
               "";
            }
            enum "ip"  {
             description
               "";
            }
         }
         description "The default forwarding mode for IGMP and MLD
Snooping is ip.
                   cisco command is as below
                   Router(config-vlan-config)# multicast snooping lookup
{ ip | mac }  ";
      }

      leaf explicit-tracking {
         if-feature explicit-tracking;
         type boolean;
         description "Tracks IGMP & MLD Snooping v3 membership reports
from individual hosts for each port of each VLAN or VSI.";
      }

      leaf exclude-lite {
          if-feature exclude-lite;
          type boolean;
          description
            "lightweight IGMPv3 and MLDv2 protocols, which simplify the
             standard versions of IGMPv3 and MLDv2.";
          reference "RFC5790";
        }

      leaf send-query {
         type boolean;
         default true;
         description "Enable quick response for topo changes.
             To support IGMP snooping in a VLAN where PIM and IGMP are
not configured.
               It cooperates with param querier-source. ";
      }

      /**
      leaf mrouter-aging-time {
         type uint16 ;
         default 180;
         description "Aging time for mrouter interface";
      }
```

```
      **/

        leaf immediate-leave {
          if-feature immediate-leave;
          type empty;
          description
            "When fast leave is enabled, the IGMP software assumes that
no more than one host is present on each VLAN port.";
        }

        leaf last-member-query-interval {
            type uint16 {
              range "1..65535";
            }
            units seconds;
            default 1;
            description
              "Last Member Query Interval, which may be tuned to modify
the
               leave latency of the network.";
             reference "RFC3376. Sec. 8.8.";
        }

       leaf query-interval {

            type uint16;
            units seconds;
            default 125;
            description
              "The Query Interval is the interval between General
Queries
               sent by the Querier.";
            reference "RFC3376. Sec. 4.1.7, 8.2, 8.14.2.";
        }

        leaf query-max-response-time {

            type uint16;
            units seconds;
            default 10;
            description
              "Query maximum response time specifies the maximum time
               allowed before sending a responding report.";
            reference "RFC3376. Sec. 4.1.1, 8.3, 8.14.3.";

        }

        leaf require-router-alert {
          if-feature require-router-alert;
          type boolean;
```

```
        default false;
          description
            "When the value is true, router alert exists in the IP head
of IGMP or MLD packet.";
        }

        leaf robustness-variable {
            type uint8 {
              range "2..7";
            }
            default 2;
            description
              "Querier's Robustness Variable allows tuning for the
expected
               packet loss on a network.";
            reference "RFC3376. Sec. 4.1.6, 8.1, 8.14.1.";
        }

        leaf version {
          type uint8 {
            range "1..3";
          }
          description "IGMP and MLD Snooping version.";
        }


      leaf-list static-bridge-mrouter-interface {

         when "../type = 'bridge'";
         if-feature static-mrouter-interface;
         type if:interface-ref;
         description "static mrouter interface in bridge fowarding";

      }

      leaf-list static-l2vpn-mrouter-interface-ac {

         when "../type = 'l2vpn'";
         if-feature static-mrouter-interface;
         type if:interface-ref;
         description "static mrouter interface whose type is interface
in l2vpn forwarding";

      }

      leaf-list static-l2vpn-mrouter-interface-pw {

         when "../type = 'l2vpn'";
         if-feature static-mrouter-interface;
         type l2vpn-instance-pw-ref;
```

```
         description "static mrouter interface whose type is pw in l2vpn
forwarding";

      }


      } // instance-config-attributes-igmp-mld-snooping

      grouping instance-config-attributes-mld-snooping {
        description "MLD snooping configuration of each VLAN.";

        uses instance-config-attributes-igmp-mld-snooping;

      leaf querier-source {
          type inet:ipv6-address;
          description
         "Use the MLD snooping querier to support MLD snooping where PIM
and MLD are not configured.
          The IP address is used as the source address in messages.";
        }

        list static-l2-multicast-group {
         if-feature static-l2-multicast-group;
         key "group source-addr";
             description
               "A static multicast route, (*,G) or (S,G).";

         leaf group {
            type inet:ipv6-address;
            description
            "Multicast group IP address";
         }

         leaf source-addr {
               type source-ipv6-addr-type;
               description
                 "Multicast source IP address.";
             }

         leaf-list bridge-outgoing-interface {
            when "../../type = 'bridge'";
            type if:interface-ref;
            description "Outgoing interface in bridge fowarding";

         }

         leaf-list l2vpn-outgoing-ac {
            when "../../type = 'l2vpn'";
            type l2vpn-instance-ac-ref;
            description "Outgoing ac in l2vpn fowarding";
```

```
         }

         leaf-list l2vpn-outgoing-pw {
            when "../../type = 'l2vpn'";
            type l2vpn-instance-pw-ref;
            description "Outgoing pw in l2vpn fowarding";

         }


       } // static-l2-multicast-group

      } // instance-config-attributes-mld-snooping

     grouping instance-state-group-attributes-igmp-mld-snooping {
        description
          "Attributes for both IGMP and MLD snooping groups.";

      leaf mac-address {
         type yang:phys-address;
         description "Destination mac address for L2 multicast
forwarding.";
      }

        leaf expire {
          type uint32;
          units seconds;
          description
         "The time left before multicast group timeout.";
        }

        leaf up-time {
          type uint32;
          units seconds;
          description
            "The time after the device created L2 multicast record.";
        }


      } // instance-state-group-attributes-igmp-mld-snooping


      grouping instance-state-attributes-igmp-snooping {

        description
          "State attributes for IGMP snooping for each VLAN or l2vpn
instance or EVPN instance.";

        uses instance-state-attributes-igmp-mld-snooping;
```

```
        list group {

        key "address";

        config false;

          description "IGMP snooping information";

          leaf address {
            type inet:ipv4-address;
            description
              "Multicast group IP address";
          }

        uses instance-state-group-attributes-igmp-mld-snooping;

        leaf last-reporter {
            type inet:ipv4-address;
            description
              "The last host address which has sent the
               report to join the multicast group.";
          }

          list source {
            key "address";
         description "Source IP address for multicast stream";
            leaf address {
              type inet:ipv4-address;
              description "Source IP address for multicast stream";
            }

            uses instance-state-source-attributes-igmp-mld-snooping;

         leaf last-reporter {
              type inet:ipv4-address;
              description
                "The last host address which has sent the
                 report to join the multicast source and group.";
            }

         list host {
              if-feature explicit-tracking;
              key "host-address";
              description
                "List of multicast membership hosts
                 of the specific multicast source-group.";

              leaf host-address {
                type inet:ipv4-address;
```

```
                description
                  "Multicast membership host address.";
              }
              leaf host-filter-mode {
                type enumeration {
                  enum "include" {
                    description
                      "In include mode";
                  }
                  enum "exclude" {
                    description
                      "In exclude mode.";
                  }
                }
                description
                  "Filter mode for a multicast membership
                   host may be either include or exclude.";
              }
            }// list host

          } // list source
        } // list group

      } // instance-state-attributes-igmp-snooping

      grouping instance-state-attributes-igmp-mld-snooping {

          description
            "State attributes for both IGMP and MLD Snooping of each
VLAN or l2vpn instance or EVPN instance.";

        leaf entries-count {
            type uint32;
         config false;
            description
              "The number of L2 multicast entries in IGMP and MLD
Snooping.";
          }

        leaf-list bridge-mrouter-interface {

         when "../type = 'bridge'";
         type if:interface-ref;
         config false;
         description " mrouter interface in bridge fowarding";

        }

        leaf-list l2vpn-mrouter-interface-ac {
```

```
          when "../type = 'l2vpn'";
          type if:interface-ref;
          config false;
          description " mrouter interface whose type is interface in
l2vpn fowarding";

         }

         leaf-list l2vpn-mrouter-interface-pw {

          when "../type = 'l2vpn'";
          type l2vpn-instance-pw-ref;
          config false;
          description " mrouter interface whose type is pw in l2vpn
fowarding";

         }


       } // instance-config-attributes-igmp-mld-snooping

       grouping instance-state-attributes-mld-snooping {
         description
           "State attributes for MLD snooping of each VLAN.";

         uses instance-state-attributes-igmp-mld-snooping;


         list group {

         key "address";

         config false;

           description "MLD snooping statistics information";

           leaf address {
             type inet:ipv6-address;
             description
               "Multicast group IP address";
           }

         uses instance-state-group-attributes-igmp-mld-snooping;

         leaf last-reporter {
             type inet:ipv6-address;
             description
               "The last host address which has sent the
                report to join the multicast group.";
           }
```

```
          list source {
            key "address";
         description "Source IP address for multicast stream";

            leaf address {
              type inet:ipv6-address;
              description "Source IP address for multicast stream";
            }

            uses instance-state-source-attributes-igmp-mld-snooping;

         leaf last-reporter {
            type inet:ipv6-address;
            description
               "The last host address which has sent the report to join
the multicast source and group.";
         }

         list host {
              if-feature explicit-tracking;
              key "host-address";
              description
                "List of multicast membership hosts
                 of the specific multicast source-group.";

              leaf host-address {
                type inet:ipv6-address;
                description
                  "Multicast membership host address.";
              }
              leaf host-filter-mode {
                type enumeration {
                  enum "include" {
                    description
                      "In include mode";
                  }
                  enum "exclude" {
                    description
                      "In exclude mode.";
                  }
                }
                description
                  "Filter mode for a multicast membership
                   host may be either include or exclude.";
              }
            }// list host

          } // list source
        } // list group
```

```
      } // instance-state-attributes-mld-snooping

      grouping instance-state-source-attributes-igmp-mld-snooping {
        description
          "State attributes for both IGMP and MLD Snooping of each VLAN
or l2vpn instance or EVPN instance.";

      leaf-list bridge-outgoing-interface {
         when "../../../type = 'bridge'";
         type if:interface-ref;
         description "Outgoing interface in bridge fowarding";

      }

      leaf-list l2vpn-outgoing-ac {
         when "../../../type = 'l2vpn'";
         type l2vpn-instance-ac-ref;
         description "Outgoing ac in l2vpn fowarding";

      }

      leaf-list l2vpn-outgoing-pw {
         when "../../../type = 'l2vpn'";
         type l2vpn-instance-pw-ref;
         description "Outgoing pw in l2vpn fowarding";

      }

        leaf up-time {
          type uint32;
          units seconds;
          description "The time after the device created L2 multicast
record";
        }

      leaf expire {
          type uint32;
          units seconds;
          description
            "The time left before multicast group timeout.";
        }

      leaf host-count {
          if-feature explicit-tracking;
          type uint32;
          description
            "The number of host addresses.";
        }
```

```
      } // instance-state-source-attributes-igmp-mld-snooping

      grouping general-statistics-error {
        description
          "A grouping defining statistics attributes for errors.";

        leaf checksum {
          type yang:counter64;
          description
            "The number of checksum errors.";
        }
        leaf too-short {
          type yang:counter64;
          description
            "The number of messages that are too short.";
        }
      } // general-statistics-error

      grouping general-statistics-sent-received {
        description
          "A grouping defining statistics attributes.";

        leaf query {
          type yang:counter64;
          description
            "The number of query messages.";
        }
        leaf membership-report-v1 {
          type yang:counter64;
          description
            "The number of membership report v1 messages.";
        }
      leaf membership-report-v2 {
          type yang:counter64;
          description
            "The number of membership report v2 messages.";
        }
      leaf membership-report-v3 {
          type yang:counter64;
          description
            "The number of membership report v3 messages.";
        }
        leaf leave {
          type yang:counter64;
          description
            "The number of leave messages.";
        }
      leaf non-member-leave {
          type yang:counter64;
          description
```

```
            "The number of non member leave messages.";
        }
      leaf pim {
          type yang:counter64;
          description
            "The number of pim hello messages.";
        }
      } // general-statistics-sent-received

   grouping interface-endpoint-attributes-igmp-snooping {

      description "interface attributes for igmp snooping";

      list host {

            if-feature explicit-tracking;

            key "host-address";

            config false;

            description
               "List of multicast membership hosts
               of the specific multicast source-group.";



            leaf host-address {
               type inet:ipv4-address;
               description
               "Multicast membership host address.";
            }
            leaf host-filter-mode {
               type enumeration {
               enum "include" {
                  description
                  "In include mode";
               }
               enum "exclude" {
                  description
                  "In exclude mode.";
               }
               }
               description
               "Filter mode for a multicast membership
               host may be either include or exclude.";
            }
      }// list host
   } // interface-endpoint-attributes-igmp-snooping
```

```
   grouping interface-endpoint-attributes-mld-snooping {

      description "interface endpoint attributes mld snooping";

      list host {

            if-feature explicit-tracking;

            key "host-address";

            config false;

            description
               "List of multicast membership hosts
               of the specific multicast source-group.";

            leaf host-address {
               type inet:ipv6-address;
               description
               "Multicast membership host address.";
            }
            leaf host-filter-mode {
               type enumeration {
               enum "include" {
                  description
                  "In include mode";
               }
               enum "exclude" {
                  description
                  "In exclude mode.";
               }
               }
               description
               "Filter mode for a multicast membership
               host may be either include or exclude.";
            }
      }// list host
   } // interface-endpoint-attributes-mld-snooping

      /*
       * igmp-snooping-instance
       */
   container igmp-snooping-instances {
       description
         "igmp-snooping-instance list";


      list igmp-snooping-instance {
       key "name";
       description
```

```
                "IGMP Snooping instance to configure the igmp-
snooping.";

       leaf name {
          type string;
          description
            "Name of the igmp-snooping-instance to configure the igmp
snooping.";
         }

       leaf type {
         type enumeration {
            enum "bridge" {
               description "bridge";
            }
            enum "l2vpn" {
               description "l2vpn";
            }
         }
         description "The type indicates bridge or l2vpn.";
       }


       uses instance-config-attributes-igmp-snooping {
           if-feature per-instance-config;
         }

       uses instance-state-attributes-igmp-snooping;

      } //igmp-snooping-instance
    } //igmp-snooping-instances




      /*
       * mld-snooping-instance
       */
   container mld-snooping-instances {
       description
         "mld-snooping-instance list";


      list mld-snooping-instance {
       key "name";
       description
                "MLD Snooping instance to configure the mld-snooping.";

       leaf name {
          type string;
```

```
          description
            "Name of the mld-snooping-instance to configure the mld
snooping.";
         }


       leaf type {
         type enumeration {
            enum "bridge" {
               description "bridge";
            }
            enum "l2vpn" {
               description "l2vpn";
            }
         }
         description "The type indicates bridge or l2vpn.";
       }


       uses instance-config-attributes-mld-snooping {
           if-feature per-instance-config;
         }

       uses instance-state-attributes-mld-snooping;

      } //mld-snooping-instance
    } //mld-snooping-instances



      container bridges {
       description
         "Apply igmp-mld-snooping instance in the bridge scenario";

         list bridge {
            key name;

            description
               "bridge list";

            leaf name {
               type name-type;
               description
               "bridge name";

            }
            leaf igmp-snooping-instance {
               type igmp-snooping-instance-ref;
               description "Configure igmp-snooping instance under the
bridge view";
```

```
            }
            leaf mld-snooping-instance {
               type mld-snooping-instance-ref;
               description "Configure mld-snooping instance under the
bridge view";
            }
            list component {
               key "name";
               description
               " ";

               leaf name {
                  type string;
                  description
                     "The name of the Component.";
               }
               container bridge-vlan {
                  description "bridge vlan";
                  list vlan {
                     key "vid";
                     description
                     "";

                     leaf vid {
                        type vlan-index-type;
                        description
                           "The VLAN identifier to which this entry
applies.";

                     }
                     leaf igmp-snooping-instance {
                        type igmp-snooping-instance-ref;
                        description "Configure igmp-snooping instance
under the vlan view";
                     }
                     leaf mld-snooping-instance {
                        type mld-snooping-instance-ref;
                        description "Configure mld-snooping instance
under the vlan view";
                     }

                  }//vlan
               }//bridge-vlan
             }//component
         }//bridge
     }   //bridges
```

```
   container l2vpn-instances {
      description "Apply igmp-mld-snooping instance in the l2vpn
scenario";

      list l2vpn-instance {
         key "name";
         description "An l2vpn service instance";

         leaf name {
               type string;
               description "Name of l2vpn service instance";
         }

         leaf igmp-snooping-instance {
            type igmp-snooping-instance-ref;
            description "Configure igmp-snooping instance under the
l2vpn-instance view";
         }
         leaf mld-snooping-instance {
            type mld-snooping-instance-ref;
            description "Configure mld-snooping instance under the
l2vpn-instance view";
         }

      }
      }

   /* augments */

   augment "/if:interfaces/if:interface" {
      description "Augment interface for referencing attributes which
only fit for interface view.";

      container igmp-mld-snooping {
         description
            "igmp-mld-snooping related attributes under interface view";

         leaf enable {
            if-feature admin-enable;
            type boolean;
            description
               "Set the value to true to enable IGMP and MLD Snooping in
the VLAN instance.";
         }

         leaf version {
            type uint8 {
               range "1..3";
            }
            description "IGMP and MLD Snooping version.";
```

```
         }

         leaf type {
            type enumeration {
               enum "bridge" {
                  description "bridge";
               }
               enum "l2vpn" {
                  description "l2vpn";
               }
            }
            description "The type indicates bridge or l2vpn.";
         }

         container static-mrouter-interface {
            description
                "Container for choice static-mrouter-interface";

            choice static-mrouter-interface {
               description
                  "Configure static multicast router interface under the
interface view";

               case bridge {
                  when "type = 'bridge'" {
                     description
                        "Applies to bridge scenario.";
                  }
                  description
                  "Applies to bridge scenario.";

                     leaf bridge-name {
                        type string;
                        description
                           "bridge name.";

                     }
                     leaf-list vlan-id {
                        type uint32;
                        description
                           "vlan id.";
                     }


               }
               case l2vpn {
                  when "type = 'l2vpn'" {
                     description
                        "Applies to l2vpn scenario.";
                  }
```

```
                  description
                  "Applies to l2vpn scenario.";

                  leaf l2vpn-instance-name {
                     type string;
                     description
                        "The l2vpn instance name applied in the
interface";

                  }
               }

            } // choice static-mrouter-interface
         }  // container static-mrouter-interface

         container static-l2-multicast-group {
            description
                "Container for static-l2-multicast-group";

            choice static-l2-multicast-group {
               description
                  "Configure static l2 multicast group under the
interface view";

               case bridge {
                  when "type = 'bridge'" {
                     description
                        "Applies to bridge scenario.";
                  }
                  description
                  "Applies to bridge scenario.";

                  leaf bridgename {
                     type string;
                     description
                        "bridge name.";

                  }

                  list bridge-group-v4 {


                     key "address";

                     description "";

                     leaf address {
                        type inet:ipv4-address;
                        description
                        "Multicast group IPV4 address";
```

```
                     }

                     leaf-list source {
                        type inet:ipv4-address;
                        description "Source IPV4 address for multicast
stream";
                     }

                     leaf-list vlan-id {
                        type uint32;
                        description
                           "vlan id.";
                     }

                  }

                  list bridge-group-v6 {
                     key "address";
                     description "";
                     leaf address {
                        type inet:ipv6-address;
                        description
                        "Multicast group IPv6 address";
                     }

                     leaf-list source {
                        type inet:ipv6-address;
                        description "Source IPv6 address for multicast
stream";
                     }

                     leaf-list vlan-id {
                        type uint32;
                        description
                           "vlan id.";
                     }

                  }

               }
               case l2vpn {

                  when "type = 'l2vpn'" {
                     description
                        "Applies to l2vpn scenario.";
                  }
                  description
                  "Applies to l2vpn scenario.";

                  list l2vpn-group-v4 {
```

```
                     key "address";
                     description "";
                     leaf address {
                        type inet:ipv4-address;
                        description
                        "Multicast group IP address";
                     }

                     leaf-list source {
                        type inet:ipv4-address;
                        description "Source IP address for multicast
stream";
                     }

                     leaf l2vpn-instance-name {
                        type string;
                        description
                           "The l2vpn instance name applied in the
interface";

                     }
                  }
                  list l2vpn-group-v6 {
                     key "address";
                     description "";

                     leaf address {
                        type inet:ipv6-address;
                        description
                        "Multicast group IP address";
                     }

                     leaf-list source {
                        type inet:ipv6-address;
                        description "Source IP address for multicast
stream";
                     }

                     leaf l2vpn-instance-name {
                        type string;
                        description
                           "The l2vpn instance name applied in the
interface";

                     }
                  }
               }

            } //choice static-l2-multicast-group
```

```
         } // container static-l2-multicast-group



         container statistics {
            config false;
            description
            "A collection of interface-related statistics objects.";

            uses general-state-attributes;

         }

      }

   }


   /*  RPCs  */

   rpc clear-igmp-snooping-groups {
        if-feature rpc-clear-groups;
        description
          "Clears the specified IGMP Snooping cache tables.";

        input {

        leaf name {
         type string;
         description
            "Name of the igmp-snooping-instance";
        }

          leaf group {
            type inet:ipv4-address;
            description
              "Multicast group IPv4 address.
               If it is not specified, all IGMP snooping group tables
are
               cleared.";
          }

          leaf source {
            type inet:ipv4-address;
            description
              "Multicast source IPv4 address.
               If it is not specified, all IGMP snooping source-group
tables are
               cleared.";
          }
```

```
        }
      } // rpc clear-igmp-snooping-groups

     rpc clear-mld-snooping-groups {
        if-feature rpc-clear-groups;
        description
          "Clears the specified MLD Snooping cache tables.";

        input {
          leaf name {
         type string;
         description
            "Name of the mld-snooping-instance";
        }

          leaf group {
            type inet:ipv6-address;
            description
              "Multicast group IPv6 address.
               If it is not specified, all MLD snooping group tables are
               cleared.";
          }

          leaf source {
            type inet:ipv6-address;
            description
              "Multicast source IPv6 address.
               If it is not specified, all MLD snooping source-group
tables are
               cleared.";
          }
        }
      } // rpc clear-mld-snooping-groups
}
<CODE ENDS>
```

4. Security Considerations

   The data model defined does not create any security implications.

5. IANA Considerations

   This draft does not request any IANA action.

6. Normative References

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[RFC6021] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6021, October 2010.

[RFC4541] M. Christensen, K. Kimball, F. Solensky, "Considerations for Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD) Snooping Switches", RFC 4541, May 2006.

[RFC2236] Fenner, W., "Internet Group Management Protocol, Version 2", RFC 2236, November 1997.

[RFC2710] Deering, S., Fenner, W., and B. Haberman, "Multicast Listener Discovery (MLD) for IPv6", RFC 2710, October 1999.

[RFC3376] Cain, B., Deering, S., Kouvelas, I., Fenner, B., and A. Thyagarajan, "Internet Group Management Protocol, Version 3", RFC 3376, October 2002.

[RFC3810] Vida, R. and L. Costa, "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", RFC 3810, June 2004.

[RFC4604] Holbrook, H., Cain, B., and B. Haberman, "Using InternetGroup Management Protocol Version 3 (IGMPv3) and Multicast Listener Discovery Protocol Version 2 (MLDv2) for Source-Specific Multicast", RFC 4604, August 2006.

[RFC4607] Holbrook, H. and B. Cain, "Source-Specific Multicast for IP", RFC 4607, August 2006.

[draft-ietf-pim-igmp-mld-yang-01] X. Liu, F. Guo, M. Sivakumar, P. McAllister, A. Peter, "A YANG data model for Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD)", draft-ietf-pim-igmp-mld-yang-01, October 28, 2016.

[draft-ietf-pim-igmp-mld-yang-03] X. Liu, F. Guo, M. Sivakumar, P. McAllister, A. Peter, "A YANG data model for Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD)", draft-ietf-pim-igmp-mld-yang-03, March 13, 2017.

[draft-dsdt-nmda-guidelines-01] M. Bjorklund, J. Schoenwaelder, P. Shafer, K. Watsen, R. Wilton, "Guidelines for YANG Module Authors (NMDA)", draft-dsdt-nmda-guidelines-01, May 2017 [draft-bjorklund-netmod-rfc7223bis-00] M. Bjorklund, "A YANG Data Model for Interface Management", draft-bjorklund-netmod-rfc7223bis-00, August 21, 2017

[draft-bjorklund-netmod-rfc7277bis-00] M. Bjorklund, "A YANG Data Model for IP Management", draft-bjorklund-netmod-rfc7277bis-00, August 21, 2017

[draft-ietf-netmod-revised-datastores-03] M. Bjorklund, J. Schoenwaelder, P. Shafer, K. Watsen, R. Wilton, "Network Management Datastore Architecture", draft-ietf-netmod-revised-datastores-03, July 3, 2017

[draft-ietf-bess-evpn-yang-02] P.Brissette, A. Sajassi, H. Shah, Z. Li, H. Chen, K. Tiruveedhula, I. Hussain, J. Rabadan, "Yang Data Model for EVPN", draft-ietf-bess-evpn-yang-02, March 13, 2017

[draft-ietf-bess-l2vpn-yang-06] H. Shah, P. Brissette, I. Chen, I. Hussain, B. Wen, K. Tiruveedhula, "YANG Data Model for MPLS-based L2VPN", draft-ietf-bess-l2vpn-yang-06.txt, June 30, 2017

Authors' Addresses

Hongji Zhao
Ericsson (China) Communications Company Ltd.
Ericsson Tower, No. 5 Lize East Street,
Chaoyang District Beijing 100102, P.R. China

Email: hongji.zhao@ericsson.com

Xufeng Liu
Jabil
8281 Greensboro Drive, Suite 200
McLean VA 22102
USA

EMail: Xufeng_Liu@jabil.com

Yisong Liu
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: liuyisong@huawei.com

Anish Peter
Individual

EMail: anish.ietf@gmail.com

Mahesh Sivakumar
Cisco Systems
510 McCarthy Boulevard
Milpitas, California
USA

EMail: masivaku@cisco.com

PIM Working Group
Internet Draft
Intended status: Standards Track
Expires: February 14, 2021

H. Zhao
Ericsson
X. Liu
Volta Networks
Y. Liu
China Mobile
M. Sivakumar
Juniper
A. Peter
Individual

August 15, 2020

A Yang Data Model for IGMP and MLD Snooping
draft-ietf-pim-igmp-mld-snooping-yang-18.txt

Abstract

This document defines a YANG data model that can be used to configure and manage Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD) Snooping devices. The YANG module in this document conforms to Network Management Datastore Architecture (NMDA).

Copyright Notice

Table of Contents

1. Introduction

This document defines a YANG [RFC7950] data model for the management of Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD) Snooping [RFC4541] devices.

The YANG module in this document conforms to the Network Management Datastore Architecture defined in [RFC8342]. The "Network Management Datastore Architecture" (NMDA) adds the ability to inspect the current operational values for configuration, allowing clients to use identical paths for retrieving the configured values and the operational values.

1.1. Terminology

The terminology for describing YANG data models is found in [RFC6020]

and [RFC7950], including:

* augment
* data model
* data node
* identity
* module

The following terminologies are used in this document:

* mrouter: multicast router, which is a router that has multicast routing enabled [RFC4286].

* mrouter interfaces: snooping switch ports where multicast routers are attached [RFC4541].

The following abbreviations are used in this document and defined model:

IGMP: Internet Group Management Protocol [RFC3376].

MLD: Multicast Listener Discovery [RFC3810].

AC: Attachment Circuit [RFC3916].

PW: Pseudo Wire [RFC3916].

1.2. Tree Diagrams

Tree diagrams used in this document follow the notation defined in

[RFC8340].

1.3. Prefixes in Data Node Names

In this document, names of data nodes, actions, and other data model objects are often used without a prefix, as long as it is clear from the context in which YANG module each name is defined. Otherwise, names are prefixed using the standard prefix associated with the corresponding YANG module, as shown in Table 1.

Prefix	YANG module	Reference
inet	ietf-inet-types	[RFC6991]
yang	ietf-yang-types	[RFC6991]
if	ietf-interfaces	[RFC8343]
rt	ietf-routing	[RFC8349]
rt-types	ietf-routing-types	[RFC8294]
ni	ietf-network-instance	[RFC8529]
pw	ietf-pseudowires	[draft-ietf-bess-l2vpn-yang]
l2vpn	ietf-l2vpn	[draft-ietf-bess-l2vpn-yang]
dot1q	ieee802-dot1q-bridge	[dot1Qcp]

Table 1: Prefixes and Corresponding YANG Modules

2. Design of Data Model

An IGMP/MLD snooping switch [RFC4541] analyzes IGMP/MLD packets and sets up forwarding tables for multicast traffic. If a switch does not run IGMP/MLD snooping, multicast traffic will be flooded in the broadcast domain. If a switch runs IGMP/MLD snooping, multicast traffic will be forwarded based on the forwarding tables to avoid wasting bandwidth. The IGMP/MLD snooping switch does not need to run any of the IGMP/MLD

protocols. Because the IGMP/MLD snooping is independent of the IGMP/MLD protocols, the data model defined in this document does not augment, or even require, the IGMP/MLD data model defined in [RFC8652].
The model covers considerations for Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD) Snooping Switches [RFC4541].

IGMP and MLD snooping switches do not adhere to the conceptual model that provides the strict separation of functionality between different communications layers in the ISO model, and instead utilize information in the upper level protocol headers as factors to be considered in processing at the lower levels [RFC4541].

IGMP Snooping switches utilize IGMP, and could support IGMPv1 [RFC1112], IGMPv2 [RFC2236], and IGMPv3 [RFC3376]. MLD Snooping switches utilize MLD, and could support MLDv1 [RFC2710] and MLDv2 [RFC3810]. The goal of this document is to define a data model that provides a common user interface to IGMP and MLD Snooping.

2.1. Overview

The IGMP and MLD Snooping YANG module defined in this document has all the common building blocks for the IGMP and MLD Snooping switches.

The YANG module includes IGMP and MLD Snooping instance definition, using instance in the L2 service type of BRIDGE [dot1Qcp] and L2VPN [draft-ietf-bess-l2vpn-yang]. The module also includes actions for clearing IGMP and MLD Snooping group tables.

2.2. Optional Capabilities

This model is designed to represent the basic capability subsets of IGMP and MLD Snooping. The main design goals of this document are that the basic capabilities described in the model are supported by any major now-existing implementation, and that the configuration of all implementations meeting the specifications is easy to express through some combination of the optional features in the model and simple vendor augmentations.

There is also value in widely supported features being standardized, to provide a standardized way to access these features, to save work for individual vendors, and so that mapping between different vendors' configuration is not needlessly complicated. Therefore, this model declares a number of features representing capabilities that not all deployed devices support.

The extensive use of feature declarations should also substantially simplify the capability negotiation process for a vendor's IGMP and MLD Snooping implementations.

On the other hand, operational state parameters are not so widely designated as features, as there are many cases where the defaulting of an operational state parameter would not cause any harm to the system, and it is much more likely that an implementation without native support for a piece of operational state would be able to derive a suitable value for a state variable that is not natively supported.

2.3. Position of Address Family in Hierarchy

IGMP Snooping only supports IPv4, while MLD Snooping only supports IPv6. The data model defined in this document can be used for both IPv4 and IPv6 address families.

This document defines IGMP Snooping and MLD Snooping as separate schema branches in the structure. The benefits are:

* The model can support IGMP Snooping (IPv4), MLD Snooping (IPv6), or both optionally and independently. Such flexibility cannot be achieved cleanly with a combined branch.

* The structure is consistent with other YANG data models such as [RFC8652], which uses separate branches for IPv4 and IPv6.

* Having separate branches for IGMP Snooping and MLD Snooping allows minor differences in their behavior to be modelled more simply and cleanly. The two branches can better support different features and node types.

3. Module Structure

This model augments the core routing data model specified in [RFC8349].

```
      +--rw routing
         +--rw router-id?
         +--rw control-plane-protocols
         |  +--rw control-plane-protocol* [type name]
         |     +--rw type
         |     +--rw name
         |     +--rw igmp-snooping-instance <= Augmented by this Model
                         ...
         |     +--rw mld-snooping-instance   <= Augmented by this Model
                         ...
```

The "igmp-snooping-instance" container instantiates an IGMP Snooping Instance. The "mld-snooping-instance" container instantiates an MLD Snooping Instance.

The YANG data model defined in this document conforms to the Network Management Datastore Architecture (NMDA) [RFC8342]. The operational state data is combined with the associated configuration data in the same hierarchy [RFC8407].

3.1. IGMP Snooping Instances

The YANG module ietf-igmp-mld-snooping augments /rt:routing/rt:control-plane-protocols/rt:control-plane-protocol to add the igmp-snooping-instance container.

All the IGMP Snooping related attributes have been defined in the igmp-snooping-instance. The read-write attributes represent configurable data. The read-only attributes represent state data.

One igmp-snooping-instance could be used in one BRIDGE [dot1Qcp] instance or L2VPN [draft-ietf-bess-l2vpn-yang] instance. One igmp-snooping-instance corresponds to one BRIDGE instance or one L2VPN instance.

The value of l2-service-type in igmp-snooping-instance is bridge or l2vpn. When it is bridge, igmp-snooping-instance will be used in the BRIDGE service. When it is l2vpn, igmp-snooping-instance will be used in the L2VPN service.

The values of bridge-mrouter-interface, l2vpn-mrouter-interface-ac, l2vpn-mrouter-interface-pw are filled by the snooping device dynamically. They are different from static-bridge-mrouter-interface, static-l2vpn-mrouter-interface-ac, and static-l2vpn-mrouter-interface-pw which are configured.

The attributes under the interfaces show the statistics of IGMP Snooping related packets.

```
  augment /rt:routing/rt:control-plane-protocols
            /rt:control-plane-protocol:
    +--rw igmp-snooping-instance {igmp-snooping}?
       +--rw l2-service-type?                    l2-service-type
       +--rw enable?                             boolean
       +--rw forwarding-table-type?              enumeration
       +--rw explicit-tracking?                  boolean
       |       {explicit-tracking}?
       +--rw lite-exclude-filter?                empty
       |       {lite-exclude-filter}?
       +--rw send-query?                         boolean
       +--rw immediate-leave?                    empty
       |       {immediate-leave}?
       +--rw last-member-query-interval?         uint16
       +--rw query-interval?                     uint16
       +--rw query-max-response-time?            uint16
       +--rw require-router-alert?               boolean
       |       {require-router-alert}?
       +--rw robustness-variable?                uint8
       +--rw static-bridge-mrouter-interface*    if:interface-ref
       |       {static-mrouter-interface}?
       +--rw static-l2vpn-mrouter-interface-ac*  if:interface-ref
       |       {static-mrouter-interface}?
```

```
       +--rw static-l2vpn-mrouter-interface-pw*   pw:pseudowire-ref
       |       {static-mrouter-interface}?
       +--rw igmp-version?                        uint8
       +--rw querier-source?                      inet:ipv4-address
       +--rw static-l2-multicast-group* [group source-addr]
       |       {static-l2-multicast-group}?
       |  +--rw group
       |  |       rt-types:ipv4-multicast-group-address
       |  +--rw source-addr
       |  |       rt-types:ipv4-multicast-source-address
       |  +--rw bridge-outgoing-interface*   if:interface-ref
       |  +--rw l2vpn-outgoing-ac*           if:interface-ref
       |  +--rw l2vpn-outgoing-pw*           pw:pseudowire-ref
       +--ro entries-count?                       yang:gauge32
       +--ro bridge-mrouter-interface*            if:interface-ref
       +--ro l2vpn-mrouter-interface-ac*          if:interface-ref
       +--ro l2vpn-mrouter-interface-pw*          pw:pseudowire-ref
       +--ro group* [address]
       |  +--ro address
       |  |       rt-types:ipv4-multicast-group-address
       |  +--ro mac-address?     yang:phys-address
       |  +--ro expire?          rt-types:timer-value-seconds16
       |  +--ro up-time          uint32
       |  +--ro last-reporter?   inet:ipv4-address
       |  +--ro source* [address]
       |     +--ro address
       |     |       rt-types:ipv4-multicast-source-address
       |     +--ro bridge-outgoing-interface*   if:interface-ref
       |     +--ro l2vpn-outgoing-ac*           if:interface-ref
       |     +--ro l2vpn-outgoing-pw*           pw:pseudowire-ref
       |     +--ro up-time                      uint32
       |     +--ro expire?
       |     |       rt-types:timer-value-seconds16
       |     +--ro host-count?                  yang:gauge32
       |     |       {explicit-tracking}?
       |     +--ro last-reporter?               inet:ipv4-address
       |     +--ro host* [host-address] {explicit-tracking}?
       |        +--ro host-address        inet:ipv4-address
       |        +--ro host-filter-mode    filter-mode-type
       +--ro interfaces
          +--ro interface* [name]
             +--ro name          if:interface-ref
             +--ro statistics
                +--ro discontinuity-time?   yang:date-and-time
                +--ro received
                |  +--ro query-count?                   yang:counter64
                |  +--ro membership-report-v1-count?   yang:counter64
                |  +--ro membership-report-v2-count?   yang:counter64
                |  +--ro membership-report-v3-count?   yang:counter64
                |  +--ro leave-count?                   yang:counter64
                |  +--ro pim-hello-count?               yang:counter64
                +--ro sent
```

```
                   +--ro query-count?                  yang:counter64
                   +--ro membership-report-v1-count?   yang:counter64
                   +--ro membership-report-v2-count?   yang:counter64
                   +--ro membership-report-v3-count?   yang:counter64
                   +--ro leave-count?                  yang:counter64
                   +--ro pim-hello-count?              yang:counter64
```

3.2. MLD Snooping Instances

The YANG module ietf-igmp-mld-snooping augments /rt:routing/rt:control-plane-protocols/rt:control-plane-protocol to add the mld-snooping-instance container. The mld-snooping-instance could be used in the BRIDGE [dot1Qcp] or L2VPN [draft-ietf-bess-l2vpn-yang] service to enable MLD Snooping.

All the MLD Snooping related attributes have been defined in the mld-snooping-instance. The read-write attributes represent configurable data. The read-only attributes represent state data.

The mld-snooping-instance has similar structure as IGMP snooping. Some of leaves are protocol related. The mld-snooping-instance uses IPv6 addresses and mld-version, while igmp-snooping-instance uses IPv4 addresses and igmp-version. Statistic counters in each of the above snooping instances are also tailored to the specific protocol type. One mld-snooping-instance could be used in one BRIDGE instance or L2VPN instance. One mld-snooping-instance corresponds to one BRIDGE instance or L2VPN instance.

The value of l2-service-type in mld-snooping-instance is bridge or l2vpn. When it is bridge, mld-snooping-instance will be used in the BRIDGE service. When it is l2vpn, mld-snooping-instance will be used in the L2VPN service.

The values of bridge-mrouter-interface, l2vpn-mrouter-interface-ac, l2vpn-mrouter-interface-pw are filled by the snooping device dynamically. They are different from static-bridge-mrouter-interface, static-l2vpn-mrouter-interface-ac, and static-l2vpn-mrouter-interface-pw which are configured.

The attributes under the interfaces show the statistics of MLD Snooping related packets.

```
augment /rt:routing/rt:control-plane-protocols
          /rt:control-plane-protocol:
  +--rw mld-snooping-instance {mld-snooping}?
       +--rw l2-service-type?                   l2-service-type
       +--rw enable?                            boolean
       +--rw forwarding-table-type?             enumeration
       +--rw explicit-tracking?                 boolean
       |       {explicit-tracking}?
       +--rw lite-exclude-filter?               empty
```

```
       |       {lite-exclude-filter}?
       +--rw send-query?                           boolean
       +--rw immediate-leave?                      empty
       |       {immediate-leave}?
       +--rw last-member-query-interval?          uint16
       +--rw query-interval?                       uint16
       +--rw query-max-response-time?             uint16
       +--rw require-router-alert?                boolean
       |       {require-router-alert}?
       +--rw robustness-variable?                 uint8
       +--rw static-bridge-mrouter-interface*     if:interface-ref
       |       {static-mrouter-interface}?
       +--rw static-l2vpn-mrouter-interface-ac*   if:interface-ref
       |       {static-mrouter-interface}?
       +--rw static-l2vpn-mrouter-interface-pw*   pw:pseudowire-ref
       |       {static-mrouter-interface}?
       +--rw mld-version?                          uint8
       +--rw querier-source?                       inet:ipv6-address
       +--rw static-l2-multicast-group* [group source-addr]
       |       {static-l2-multicast-group}?
       |  +--rw group
       |  |       rt-types:ipv6-multicast-group-address
       |  +--rw source-addr
       |  |       rt-types:ipv6-multicast-source-address
       |  +--rw bridge-outgoing-interface*   if:interface-ref
       |  +--rw l2vpn-outgoing-ac*           if:interface-ref
       |  +--rw l2vpn-outgoing-pw*           pw:pseudowire-ref
       +--ro entries-count?                        yang:gauge32
       +--ro bridge-mrouter-interface*             if:interface-ref
       +--ro l2vpn-mrouter-interface-ac*           if:interface-ref
       +--ro l2vpn-mrouter-interface-pw*           pw:pseudowire-ref
       +--ro group* [address]
       |  +--ro address
       |  |       rt-types:ipv6-multicast-group-address
       |  +--ro mac-address?     yang:phys-address
       |  +--ro expire?          rt-types:timer-value-seconds16
       |  +--ro up-time          uint32
       |  +--ro last-reporter?   inet:ipv6-address
       |  +--ro source* [address]
       |     +--ro address
       |     |       rt-types:ipv6-multicast-source-address
       |     +--ro bridge-outgoing-interface*   if:interface-ref
       |     +--ro l2vpn-outgoing-ac*           if:interface-ref
       |     +--ro l2vpn-outgoing-pw*           pw:pseudowire-ref
       |     +--ro up-time                      uint32
       |     +--ro expire?
       |     |       rt-types:timer-value-seconds16
       |     +--ro host-count?                  yang:gauge32
       |     |       {explicit-tracking}?
       |     +--ro last-reporter?               inet:ipv6-address
       |     +--ro host* [host-address] {explicit-tracking}?
       |        +--ro host-address        inet:ipv6-address
```

```
       |        +--ro host-filter-mode    filter-mode-type
       +--ro interfaces
          +--ro interface* [name]
             +--ro name          if:interface-ref
             +--ro statistics
                +--ro discontinuity-time?   yang:date-and-time
                +--ro received
                |  +--ro query-count?       yang:counter64
                |  +--ro report-v1-count?   yang:counter64
                |  +--ro report-v2-count?   yang:counter64
                |  +--ro done-count?        yang:counter64
                |  +--ro pim-hello-count?   yang:counter64
                +--ro sent
                   +--ro query-count?       yang:counter64
                   +--ro report-v1-count?   yang:counter64
                   +--ro report-v2-count?   yang:counter64
                   +--ro done-count?        yang:counter64
                   +--ro pim-hello-count?   yang:counter64
```

3.3. Using IGMP and MLD Snooping Instances

The igmp-snooping-instance could be used in the service of BRIDGE [dot1Qcp] or L2VPN [draft-ietf-bess-l2vpn-yang] to configure the IGMP Snooping.

For the BRIDGE service this model augments /dot1q:bridges/dot1q:bridge to use igmp-snooping-instance. It means IGMP Snooping is enabled in the whole bridge.

It also augments /dot1q:bridges/dot1q:bridge/dot1q:component/ dot1q:bridge-vlan/dot1q:vlan to use igmp-snooping-instance. It means IGMP Snooping is enabled in the specified VLAN on the bridge.

```
     augment /dot1q:bridges/dot1q:bridge:
       +--rw igmp-snooping-instance?   igmp-mld-snooping-instance-ref
       +--rw mld-snooping-instance?    igmp-mld-snooping-instance-ref

     augment /dot1q:bridges/dot1q:bridge/dot1q:component
              /dot1q:bridge-vlan/dot1q:vlan:
       +--rw igmp-snooping-instance?   igmp-mld-snooping-instance-ref
       +--rw mld-snooping-instance?    igmp-mld-snooping-instance-ref
```

For the L2VPN service this model augments /ni:network-instances/ ni:network-instance/ni:ni-type/l2vpn:l2vpn [RFC8529] to use igmp-snooping-instance. It means IGMP Snooping is enabled in the specified l2vpn instance.

```
     augment /ni:network-instances/ni:network-instance/ni:ni-type
               /l2vpn:l2vpn:
       +--rw igmp-snooping-instance?   igmp-mld-snooping-instance-ref
       +--rw mld-snooping-instance?    igmp-mld-snooping-instance-ref
```

The mld-snooping-instance could be used in concurrence with igmp-snooping-instance to configure the MLD Snooping.

3.4. IGMP and MLD Snooping Actions

IGMP and MLD Snooping actions clear the specified IGMP and MLD Snooping group tables. If both source X and group Y are specified, only source X from group Y in that specific instance will be cleared.

```
augment /rt:routing/rt:control-plane-protocols
            /rt:control-plane-protocol:
    +--rw igmp-snooping-instance {igmp-snooping}?
      +---x clear-igmp-snooping-groups {action-clear-groups}?
          +---w input
             +---w group     union
             +---w source    rt-types:ipv4-multicast-source-address

augment /rt:routing/rt:control-plane-protocols
            /rt:control-plane-protocol:
    +--rw mld-snooping-instance {mld-snooping}?
       +---x clear-mld-snooping-groups {action-clear-groups}?
          +---w input
             +---w group     union
             +---w source    rt-types:ipv6-multicast-source-address
```

4. IGMP and MLD Snooping YANG Module

This module references [RFC1112],[RFC2236],[RFC2710],[RFC3376], [RFC3810],[RFC4541],[RFC5790],[RFC6636],[RFC6991],[RFC7761], [RFC8343],[RFC8529],[dot1Qcp], and [draft-ietf-bess-l2vpn-yang].

```
<CODE BEGINS> file ietf-igmp-mld-snooping@2020-08-07.yang
module ietf-igmp-mld-snooping {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-igmp-mld-snooping";

  prefix ims;

  import ietf-inet-types {
    prefix "inet";
    reference
      "RFC 6991: Common YANG Data Types";
  }
```

```
  import ietf-yang-types {
    prefix "yang";
    reference
      "RFC 6991: Common YANG Data Types";
  }

  import ietf-interfaces {
    prefix "if";
    reference
      "RFC 8343: A YANG Data Model for Interface Management";
  }

  import ietf-routing {
    prefix "rt";
    reference
      "RFC 8349: A YANG Data Model for Routing Management (NMDA
       Version)";
  }

  import ietf-routing-types {
    prefix "rt-types";
    reference
      "RFC 8294: Common YANG Data Types for the Routing Area";
  }

  import ietf-l2vpn {
    prefix "l2vpn";
    reference
      "draft-ietf-bess-l2vpn-yang: YANG Data Model for MPLS-based
L2VPN";
  }

  import ietf-network-instance {
    prefix "ni";
    reference
      "RFC 8529: YANG Data Model for Network Instances";
  }

  import ietf-pseudowires {
    prefix "pw";
    reference
      "draft-ietf-bess-l2vpn-yang: YANG Data Model for MPLS-based
L2VPN";
  }

  import ieee802-dot1q-bridge {
    prefix "dot1q";
    reference
      "dot1Qcp: IEEE 802.1Qcp-2018 Bridges and Bridged Networks
                - Amendment: YANG Data Model";
  }
```

```
  organization
    "IETF PIM Working Group";

  contact
    "WG Web:   <http://tools.ietf.org/wg/pim/>
     WG List:  <mailto:pim@ietf.org>

     Editors:  Hongji Zhao
               <mailto:hongji.zhao@ericsson.com>

               Xufeng Liu
               <mailto:xufeng.liu.ietf@gmail.com>

               Yisong Liu
               <mailto:liuyisong@chinamobile.com>

               Anish Peter
               <mailto:anish.ietf@gmail.com>

               Mahesh Sivakumar
               <mailto:sivakumar.mahesh@gmail.com>

    ";

  description
    "The module defines a collection of YANG definitions common for
     all devices that implement Internet Group Management Protocol
     (IGMP) and Multicast Listener Discovery (MLD) Snooping which is
     described in RFC 4541.

     Copyright (c) 2020 IETF Trust and the persons identified as
     authors of the code.  All rights reserved.

     Redistribution and use in source and binary forms, with or
     without modification, is permitted pursuant to, and subject to
     the license terms contained in, the Simplified BSD License set
     forth in Section 4.c of the IETF Trust’s Legal Provisions
     Relating to IETF Documents
     (http://trustee.ietf.org/license-info).

     This version of this YANG module is part of RFC XXXX; see the
     RFC itself for full legal notices.";

  revision 2020-08-07 {
    description
      "Initial revision.";
    reference
      "RFC XXXX: A YANG Data Model for IGMP and MLD Snooping";
  }
```

```
  /*
   * Features
   */

  feature igmp-snooping {
    description
      "Support IGMP snooping.";
    reference
      "RFC 4541";
  }

  feature mld-snooping {
    description
      "Support MLD snooping.";
    reference
      "RFC 4541";
  }

  feature immediate-leave {
    description
      "Support configuration of fast leave. The fast leave feature
       does not send last member query messages to hosts.";
    reference
      "RFC 3376";
  }

  feature static-l2-multicast-group {
    description
      "Support configuration of L2 multicast static-group.";
  }

  feature static-mrouter-interface {
    description
      "Support multicast router interface explicitly configured
       by management";
    reference
      "RFC 4541";
  }

  feature action-clear-groups {
    description
      "Support clearing statistics by action for IGMP & MLD snooping.";
  }

  feature require-router-alert {
    description
      "Support configuration of require-router-alert.";
    reference
      "RFC 3376";
  }

  feature lite-exclude-filter {
```

```
    description
      "Enable the support of the simplified EXCLUDE filter.";
    reference
      "RFC 5790";
  }

  feature explicit-tracking {
    description
      "Support configuration of per instance explicit-tracking.";
    reference
      "RFC 6636";
  }

  /* identities */

  identity l2-service-type {
    description
      "Base identity for L2 service type in IGMP & MLD snooping";
  }

  identity bridge {
    base l2-service-type;
    description
      "This identity represents BRIDGE service.";
  }

  identity l2vpn {
    base l2-service-type;
    description
      "This identity represents L2VPN service.";
  }

  identity filter-mode {
    description
      "Base identity for filter mode in IGMP & MLD snooping";
  }

  identity include {
    base filter-mode;
    description
      "This identity represents include mode.";
  }

  identity exclude {
    base filter-mode;
    description
      "This identity represents exclude mode.";
  }

  identity igmp-snooping {
    base rt:control-plane-protocol;
    description
```

```
      "IGMP snooping";
  }

  identity mld-snooping {
    base rt:control-plane-protocol;
    description
      "MLD snooping";
  }

  /*
   * Typedefs
   */

  typedef l2-service-type {
    type identityref {
      base "l2-service-type";
    }
    description "The L2 service type used with IGMP & MLD snooping ";
  }

  typedef filter-mode-type {
    type identityref {
      base "filter-mode";
    }
    description "The host filter mode";
  }

  typedef igmp-mld-snooping-instance-ref {
    type leafref {
      path "/rt:routing/rt:control-plane-protocols"+
        "/rt:control-plane-protocol/rt:name";
    }
    description
      "This type is used by data models which need to
       reference IGMP & MLD snooping instance.";
  }

  /*
   * Groupings
   */


  grouping instance-config-attributes-igmp-mld-snooping {
    description
      "IGMP and MLD snooping configuration of each VLAN.";

    leaf enable {
      type boolean;
      default false;
        description
          "Set the value to true to enable IGMP & MLD snooping.";
    }
```

```
    leaf forwarding-table-type {
      type enumeration {
        enum "mac" {
          description
          "MAC-based lookup mode";
        }
        enum "ip"  {
          description
          "IP-based lookup mode";
        }
      }
      default "ip";
      description "The default forwarding table type is ip";
    }

    leaf explicit-tracking {
      if-feature explicit-tracking;
      type boolean;
      default false;
      description
        "Track the IGMPv3 and MLDv2 snooping membership reports
         from individual hosts. It contributes to saving network
         resources and shortening leave latency.";
    }

    leaf lite-exclude-filter {
      if-feature lite-exclude-filter;
      type empty;
      description
        "For IGMP Snooping, the presence of this
         leaf enables the support of the simplified EXCLUDE filter
         in the Lightweight IGMPv3 protocol, which simplifies the
         standard versions of IGMPv3.
         For MLD Snooping, the presence of this
         leaf enables the support of the simplified EXCLUDE filter
         in the Lightweight MLDv2 protocol, which simplifies the
         standard versions of MLDv2.";
      reference
        "RFC 5790";
    }

    leaf send-query {
      type boolean;
      default false;
      description
        "Enable quick response for topology changes.
         To support IGMP snooping in a VLAN where PIM and IGMP are
         not configured. It cooperates with parameter querier-source.";
    }

    leaf immediate-leave {
```

```
      if-feature immediate-leave;
      type empty;
      description
        "When immediate leave is enabled, the IGMP software assumes
         that no more than one host is present on each VLAN port.";
    }

    leaf last-member-query-interval {
      type uint16 {
        range "10..10230";
      }
      units deciseconds;
      default 10;
      description
        "Last Member Query Interval, which may be tuned to modify
         the leave latency of the network.
         It is represented in units of 1/10 second.";
      reference "RFC 3376. Sec. 8.8.";
    }

    leaf query-interval {
      type uint16;
      units seconds;
      default 125;
      description
        "The Query Interval is the interval between General Queries
         sent by the Querier.";
      reference "RFC 3376. Sec. 4.1.7, 8.2, 8.14.2.";
    }

    leaf query-max-response-time {
      type uint16;
      units deciseconds;
      default 100;
      description
        "Query maximum response time specifies the maximum time
         allowed before sending a responding report.
         It is represented in units of 1/10 second.";
      reference "RFC 3376. Sec. 4.1.1, 8.3, 8.14.3.";
    }

    leaf require-router-alert {
      if-feature require-router-alert;
      type boolean;
      default false;
      description
        "When the value is true, router alert should exist
         in the IP header of IGMP or MLD packet.";
    }

    leaf robustness-variable {
      type uint8 {
```

```
        range "1..7";
      }
      default 2;
      description
        "Querier's Robustness Variable allows tuning for the
         expected packet loss on a network.";
      reference "RFC 3376. Sec. 4.1.6, 8.1, 8.14.1.";
    }

    leaf-list static-bridge-mrouter-interface {
      when 'derived-from-or-self(../l2-service-type,"ims:bridge")';
      if-feature static-mrouter-interface;
      type if:interface-ref;
      description "static mrouter interface in BRIDGE forwarding";
    }

    leaf-list static-l2vpn-mrouter-interface-ac {
      when 'derived-from-or-self(../l2-service-type,"ims:l2vpn")';
      if-feature static-mrouter-interface;
      type if:interface-ref;
      description
        "static mrouter interface whose type is interface
         in L2VPN forwarding";
    }

    leaf-list static-l2vpn-mrouter-interface-pw {
      when 'derived-from-or-self(../l2-service-type,"ims:l2vpn")';
      if-feature static-mrouter-interface;
      type pw:pseudowire-ref;
      description
        "static mrouter interface whose type is PW
         in L2VPN forwarding";
    }
  } // instance-config-attributes-igmp-mld-snooping

  grouping instance-state-group-attributes-igmp-mld-snooping {
    description
      "Attributes for both IGMP and MLD snooping groups.";

    leaf mac-address {
      type yang:phys-address;
      description "Destination MAC address for L2 multicast.";
    }

    leaf expire {
      type rt-types:timer-value-seconds16;
      units seconds;
      description
        "The time left before multicast group timeout.";
    }

    leaf up-time {
```

```
      type uint32;
      units seconds;
      mandatory true;
      description
        "The time elapsed since L2 multicast record created.";
    }
  } // instance-state-group-attributes-igmp-mld-snooping

  grouping instance-state-attributes-igmp-mld-snooping {

    description
      "State attributes for IGMP & MLD snooping instance.";

    leaf entries-count {
      type yang:gauge32;
      config false;
      description
        "The number of L2 multicast entries in IGMP & MLD snooping";
    }

    leaf-list bridge-mrouter-interface {
      when 'derived-from-or-self(../l2-service-type,"ims:bridge")';
      type if:interface-ref;
      config false;
      description
        "Indicates a list of mrouter interfaces dynamicly learned in a
         bridge. When this switch receives IGMP/MLD queries from a
         multicast router on an interface, the interface will become
         mrouter interface for IGMP/MLD snooping.";
    }

    leaf-list l2vpn-mrouter-interface-ac {
      when 'derived-from-or-self(../l2-service-type,"ims:l2vpn")';
      type if:interface-ref;
      config false;
      description
        "The mrouter interface whose type is interface in L2VPN
         forwarding. When switch receives IGMP/MLD queries from
         multicast router on an interface, this interface will
         become mrouter interface for IGMP/MLD snooping.";
    }

    leaf-list l2vpn-mrouter-interface-pw {
      when 'derived-from-or-self(../l2-service-type,"ims:l2vpn")';
      type pw:pseudowire-ref;
      config false;
      description
        "The mrouter interface whose type is PW in L2VPN forwarding.
         When switch receives IGMP/MLD queries from multicast router
         on a PW, this PW will become mrouter interface for IGMP/MLD
         snooping.";
    }
```

```
  } // instance-config-attributes-igmp-mld-snooping


  grouping instance-state-source-attributes-igmp-mld-snooping {
    description
      "State attributes for IGMP & MLD snooping instance.";

    leaf-list bridge-outgoing-interface {
      when 'derived-from-or-self(../../../l2-service-
type,"ims:bridge")';
      type if:interface-ref;
      description "Outgoing interface in BRIDGE forwarding";
    }

    leaf-list l2vpn-outgoing-ac {
      when 'derived-from-or-self(../../../l2-service-type,"ims:l2vpn")';
      type if:interface-ref;
      description "Outgoing Attachment Circuit (AC) in L2VPN";
    }

    leaf-list l2vpn-outgoing-pw {
      when 'derived-from-or-self(../../../l2-service-type,"ims:l2vpn")';
      type pw:pseudowire-ref;
      description "Outgoing Pseudo Wire (PW) in L2VPN";
    }

    leaf up-time {
      type uint32;
      units seconds;
      mandatory true;
      description
        "The time elapsed since L2 multicast record created";
    }

    leaf expire {
      type rt-types:timer-value-seconds16;
      units seconds;
      description
        "The time left before multicast group timeout.";
    }

    leaf host-count {
      if-feature explicit-tracking;
      type yang:gauge32;
      description
        "The number of host addresses.";
    }
  } // instance-state-source-attributes-igmp-mld-snooping

  grouping igmp-snooping-statistics {
    description
      "The statistics attributes for IGMP snooping.";
```

```
      leaf query-count {
        type yang:counter64;
        description
          "The number of Membership Query messages.";
        reference
          "RFC 2236";
      }
      leaf membership-report-v1-count {
        type yang:counter64;
        description
          "The number of Version 1 Membership Report messages.";
        reference
          "RFC 1112";
      }
      leaf membership-report-v2-count {
        type yang:counter64;
        description
          "The number of Version 2 Membership Report messages.";
        reference
          "RFC 2236";
      }
      leaf membership-report-v3-count {
        type yang:counter64;
        description
          "The number of Version 3 Membership Report messages.";
        reference
          "RFC 3376";
      }
      leaf leave-count {
        type yang:counter64;
        description
          "The number of Leave Group messages.";
        reference
          "RFC 2236";
      }
      leaf pim-hello-count {
        type yang:counter64;
        description
          "The number of PIM hello messages.";
        reference
          "RFC 7761";
      }
  } // igmp-snooping-statistics

  grouping mld-snooping-statistics {
    description
      "The statistics attributes for MLD snooping.";

      leaf query-count {
        type yang:counter64;
        description
```

```
          "The number of Multicast Listener Query messages.";
        reference
          "RFC 3810";
      }
      leaf report-v1-count {
        type yang:counter64;
        description
          "The number of Version 1 Multicast Listener Report.";
        reference
          "RFC 2710";
      }
      leaf report-v2-count {
        type yang:counter64;
        description
          "The number of Version 2 Multicast Listener Report.";
        reference
          "RFC 3810";
      }
      leaf done-count {
        type yang:counter64;
        description
          "The number of Version 1 Multicast Listener Done.";
        reference
          "RFC 2710";
      }
      leaf pim-hello-count {
        type yang:counter64;
        description
          "The number of PIM hello messages.";
        reference
          "RFC 7761";
      }
  } // mld-snooping-statistics

  augment "/rt:routing/rt:control-plane-protocols"+
        "/rt:control-plane-protocol" {
    when 'derived-from-or-self(rt:type, "ims:igmp-snooping")' {
      description
        "This container is only valid for IGMP snooping.";
    }
    description
      "IGMP snooping augmentation to control plane protocol
       configuration and state.";

    container igmp-snooping-instance {
      if-feature igmp-snooping;
      description
        "IGMP snooping instance to configure igmp-snooping.";

      leaf l2-service-type {
        type l2-service-type;
        default bridge;
```

```
        description
          "The l2-service-type indicates BRIDGE or L2VPN.";
      }

      uses instance-config-attributes-igmp-mld-snooping;

      leaf igmp-version {
        type uint8 {
          range "1..3";
        }
        default 2;
        description "IGMP version.";
      }

      leaf querier-source {
        type inet:ipv4-address;
        description
          "Use the IGMP snooping querier to support IGMP
           snooping in a VLAN where PIM and IGMP are not configured.
           The IPv4 address is used as source address in messages.";
      }

      list static-l2-multicast-group {
        if-feature static-l2-multicast-group;
        key "group source-addr";
        description
          "A static multicast route, (*,G) or (S,G).";

        leaf group {
          type rt-types:ipv4-multicast-group-address;
          description
            "Multicast group IPv4 address";
        }

        leaf source-addr {
          type rt-types:ipv4-multicast-source-address;
          description
            "Multicast source IPv4 address.";
        }

        leaf-list bridge-outgoing-interface {
          when 'derived-from-or-self(../../l2-service-
type,"ims:bridge")';
          type if:interface-ref;
          description "Outgoing interface in BRIDGE forwarding";
        }

        leaf-list l2vpn-outgoing-ac {
          when 'derived-from-or-self(../../l2-service-
type,"ims:l2vpn")';
          type if:interface-ref;
          description "Outgoing Attachment Circuit (AC) in L2VPN";
```

```
        }

        leaf-list l2vpn-outgoing-pw {
          when 'derived-from-or-self(../../l2-service-
type,"ims:l2vpn")';
          type pw:pseudowire-ref;
          description "Outgoing Pseudo Wire (PW) in L2VPN";
        }
      } // static-l2-multicast-group

      uses instance-state-attributes-igmp-mld-snooping;

      list group {

        key "address";

        config false;

        description "IGMP snooping information";

        leaf address {
          type rt-types:ipv4-multicast-group-address;
          description
            "Multicast group IPv4 address";
        }

        uses instance-state-group-attributes-igmp-mld-snooping;

        leaf last-reporter {
          type inet:ipv4-address;
          description
            "Address of the last host which has sent report to join
             the multicast group.";
        }

        list source {
          key "address";
          description "Source IPv4 address for multicast stream";

          leaf address {
            type rt-types:ipv4-multicast-source-address;
            description "Source IPv4 address for multicast stream";
          }

          uses instance-state-source-attributes-igmp-mld-snooping;

          leaf last-reporter {
            type inet:ipv4-address;
            description
              "Address of the last host which has sent report
               to join the multicast group.";
          }
```

```
          list host {
            if-feature explicit-tracking;
            key "host-address";
            description
              "List of multicast membership hosts
               of the specific multicast source-group.";

            leaf host-address {
              type inet:ipv4-address;
              description
                "Multicast membership host address.";
            }
            leaf host-filter-mode {
              type filter-mode-type;
              mandatory true;
              description
                "Filter mode for a multicast membership
                 host may be either include or exclude.";
            }
          }// list host

        } // list source
      } // list group

      container interfaces {
         config false;

         description
           "Contains the interfaces associated with the IGMP snooping
            instance";

         list interface {
           key "name";

           description
             "A list of interfaces  associated with the IGMP snooping
              instance";

           leaf name {
             type if:interface-ref;
             description
               "The name of interface";

           }

           container statistics {
             description
               "The interface statistics for IGMP snooping";

             leaf discontinuity-time {
               type yang:date-and-time;
```

```
                description
                  "The time on the most recent occasion at which any one
                   or more of the statistic counters suffered a
                   discontinuity. If no such discontinuities have
                   occurred since the last re-initialization of the local
                   management subsystem, then this node contains the time
                   the local management subsystem re-initialized
itself.";
              }
              container received {
                description
                  "Number of received snooped IGMP packets";

                uses igmp-snooping-statistics;
              }
              container sent {
                description
                  "Number of sent snooped IGMP packets";

                uses igmp-snooping-statistics;
              }
            }
          }
        }

        action clear-igmp-snooping-groups {
          if-feature action-clear-groups;
          description
            "Clear IGMP snooping cache tables.";

          input {
            leaf group {
              type union {
                type enumeration {
                  enum 'all-groups' {
                    description
                      "All multicast group addresses.";
                  }
                }
                type rt-types:ipv4-multicast-group-address;
              }
              mandatory true;
              description
                "Multicast group IPv4 address. If value 'all-groups' is
                 specified, all IGMP snooping group entries are cleared
                 for specified source address.";
            }
            leaf source {
              type rt-types:ipv4-multicast-source-address;
              mandatory true;
              description
                "Multicast source IPv4 address. If value '*' is specified,
```

```
               all IGMP snooping source-group tables are cleared.";
          }
        }
      } // action clear-igmp-snooping-groups
    } // igmp-snooping-instance
  } // augment

  augment "/rt:routing/rt:control-plane-protocols"+
        "/rt:control-plane-protocol" {
    when 'derived-from-or-self(rt:type, "ims:mld-snooping")' {
        description
          "This container is only valid for MLD snooping.";
    }
    description
      "MLD snooping augmentation to control plane protocol
       configuration and state.";

    container mld-snooping-instance {
      if-feature mld-snooping;
      description
        "MLD snooping instance to configure mld-snooping.";

      leaf l2-service-type {
        type l2-service-type;
        default bridge;
        description
          "The l2-service-type indicates BRIDGE or L2VPN.";
      }

      uses instance-config-attributes-igmp-mld-snooping;

      leaf mld-version {
        type uint8 {
          range "1..2";
        }
        default 2;
        description "MLD version.";
      }

      leaf querier-source {
        type inet:ipv6-address;
        description
          "Use the MLD snooping querier to support MLD snooping where
           PIM and MLD are not configured. The IPv6 address is used as
           the source address in messages.";
      }

      list static-l2-multicast-group {
        if-feature static-l2-multicast-group;
        key "group source-addr";
        description
          "A static multicast route, (*,G) or (S,G).";
```

```
        leaf group {
          type rt-types:ipv6-multicast-group-address;
          description
            "Multicast group IPv6 address";
        }

        leaf source-addr {
          type rt-types:ipv6-multicast-source-address;
          description
            "Multicast source IPv6 address.";
        }

        leaf-list bridge-outgoing-interface {
          when 'derived-from-or-self(../../l2-service-
type,"ims:bridge")';
          type if:interface-ref;
          description "Outgoing interface in BRIDGE forwarding";
        }

        leaf-list l2vpn-outgoing-ac {
          when 'derived-from-or-self(../../l2-service-
type,"ims:l2vpn")';
          type if:interface-ref;
          description "Outgoing Attachment Circuit (AC) in L2VPN";
        }

        leaf-list l2vpn-outgoing-pw {
          when 'derived-from-or-self(../../l2-service-
type,"ims:l2vpn")';
          type pw:pseudowire-ref;
          description "Outgoing Pseudo Wire (PW) in L2VPN";
        }
      } // static-l2-multicast-group

      uses instance-state-attributes-igmp-mld-snooping;

      list group {
        key "address";
        config false;
        description "MLD snooping statistics information";

        leaf address {
          type rt-types:ipv6-multicast-group-address;
          description
            "Multicast group IPv6 address";
        }

        uses instance-state-group-attributes-igmp-mld-snooping;

        leaf last-reporter {
          type inet:ipv6-address;
```

```
          description
            "Address of the last host which has sent report
             to join the multicast group.";
        }

        list source {
          key "address";
          description "Source IPv6 address for multicast stream";

          leaf address {
            type rt-types:ipv6-multicast-source-address;
            description "Source IPv6 address for multicast stream";
          }

          uses instance-state-source-attributes-igmp-mld-snooping;

          leaf last-reporter {
           type inet:ipv6-address;
           description
            "Address of the last host which has sent report
             to join the multicast group.";
          }

          list host {
            if-feature explicit-tracking;
            key "host-address";
            description
              "List of multicast membership hosts
               of the specific multicast source-group.";

            leaf host-address {
              type inet:ipv6-address;
              description
                "Multicast membership host address.";
            }
            leaf host-filter-mode {
              type filter-mode-type;
              mandatory true;
              description
                "Filter mode for a multicast membership
                 host may be either include or exclude.";
            }
          }// list host
        } // list source
      } // list group

      container interfaces {
         config false;

         description
           "Contains the interfaces associated with the MLD snooping
            instance";
```

```
         list interface {
           key "name";

           description
             "A list of interfaces associated with the MLD snooping
              instance";

           leaf name {
             type if:interface-ref;
             description
               "The name of interface";

           }

           container statistics {
             description
               "The interface statistics for MLD snooping";

             leaf discontinuity-time {
               type yang:date-and-time;
               description
                 "The time on the most recent occasion at which any one
                  or more of the statistic counters suffered a
                  discontinuity. If no such discontinuities have
                  occurred since the last re-initialization of the local
                  management subsystem, then this node contains the time
                  the local management subsystem re-initialized
itself.";
             }
             container received {
               description
                 "Number of received snooped MLD packets";

               uses mld-snooping-statistics;
             }
             container sent {
               description
                 "Number of sent snooped MLD packets";

               uses mld-snooping-statistics;
             }
           }
         }
      }

      action clear-mld-snooping-groups {
        if-feature action-clear-groups;
        description
          "Clear MLD snooping cache tables.";

        input {
```

```
          leaf group {
            type union {
              type enumeration {
                enum 'all-groups' {
                  description
                    "All multicast group addresses.";
                }
              }
              type rt-types:ipv6-multicast-group-address;
            }
            mandatory true;
            description
              "Multicast group IPv6 address. If value 'all-groups' is
               specified, all MLD snooping group entries are cleared
               for specified source address.";
          }
          leaf source {
            type rt-types:ipv6-multicast-source-address;
            mandatory true;
            description
              "Multicast source IPv6 address. If value '*' is specified,
               all MLD snooping source-group tables are cleared.";
          }
        }
      } // action clear-mld-snooping-groups
    }// mld-snooping-instance
  } // augment

  augment "/dot1q:bridges/dot1q:bridge" {
    description
      "Use IGMP & MLD snooping instance in BRIDGE.";

    leaf igmp-snooping-instance {
      type igmp-mld-snooping-instance-ref;
      description
        "Configure IGMP snooping instance under bridge view";
    }

    leaf mld-snooping-instance {
      type igmp-mld-snooping-instance-ref;
      description
        "Configure MLD snooping instance under bridge view";
    }
  }

  augment "/dot1q:bridges/dot1q:bridge"+
    "/dot1q:component/dot1q:bridge-vlan/dot1q:vlan" {
    description
      "Use IGMP & MLD snooping instance in certain VLAN of BRIDGE";

    leaf igmp-snooping-instance {
      type igmp-mld-snooping-instance-ref;
```

```
      description
        "Configure IGMP snooping instance under VLAN view";
    }

    leaf mld-snooping-instance {
      type igmp-mld-snooping-instance-ref;
      description
        "Configure MLD snooping instance under VLAN view";
    }
  }

  augment "/ni:network-instances/ni:network-instance"+
    "/ni:ni-type/l2vpn:l2vpn" {

    description
      "Use IGMP & MLD snooping instance in L2VPN.";

    leaf igmp-snooping-instance {
      type igmp-mld-snooping-instance-ref;
      description
        "Configure IGMP snooping instance in L2VPN.";
    }

    leaf mld-snooping-instance {
      type igmp-mld-snooping-instance-ref;
      description
        "Configure MLD snooping instance in L2VPN.";
    }
  }
}
<CODE ENDS>
```

5. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

The Network Configuration Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

There are a number of data nodes defined in this YANG module that are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability:

Under /rt:routing/rt:control-plane-protocols/rt:control-plane-protocol:/

ims:igmp-snooping-instance

ims:mld-snooping-instance

The subtrees under /dot1q:bridges/dot1q:bridge

ims:igmp-snooping-instance

ims:mld-snooping-instance

The subtrees under /dot1q:bridges/dot1q:bridge/dot1q:component /dot1q:bridge-vlan/dot1q:vlan

ims:igmp-snooping-instance

ims:mld-snooping-instance

Unauthorized access to any data node of these subtrees can adversely affect the IGMP & MLD Snooping subsystem of both the local device and the network. This may lead to network malfunctions, delivery of packets to inappropriate destinations, and other problems.

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

Under /rt:routing/rt:control-plane-protocols/rt:control-plane-protocol:/

ims:igmp-snooping-instance

ims:mld-snooping-instance

Unauthorized access to any data node of these subtrees can disclose the operational state information of IGMP & MLD Snooping on this device. The group/source/host information may expose multicast group memberships, and transitively the associations between the user on the host and the contents from the source which could be privately sensitive. Some of the action operations in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control access to these operations. These are the operations and their sensitivity/vulnerability:

Under /rt:routing/rt:control-plane-protocols/rt:control-plane-protocol:/

ims:igmp-snooping-instance/ims:clear-igmp-snooping-groups

ims:mld-snooping-instance/ims:clear-mld-snooping-groups

Some of the actions in this YANG module may be considered sensitive or vulnerable in some network environments. The IGMP & MLD Snooping YANG module supports the "clear-igmp-snooping-groups" and "clear-mld-snooping-groups" actions. If unauthorized action is invoked, the IGMP and MLD Snooping group tables will be cleared unexpectedly. Especially when using wildcard, all the multicast traffic will be flooded in the broadcast domain. The devices that use this YANG module should heed the Security Considerations in [RFC4541].

6. IANA Considerations

RFC Ed.: In this section, replace all occurrences of 'XXXX' with the actual RFC number (and remove this note).

6.1. XML Registry

This document registers the following namespace URIs in the IETF XML

registry [RFC3688]:

```
--------------------------------------------------------------------
URI: urn:ietf:params:xml:ns:yang:ietf-igmp-mld-snooping
Registrant Contact: The IETF.
XML: N/A, the requested URI is an XML namespace.
--------------------------------------------------------------------
```

6.2. YANG Module Names Registry

This document registers the following YANG modules in the YANG Module Names registry [RFC7950]:

```
--------------------------------------------------------------------
name:         ietf-igmp-mld-snooping
namespace:    urn:ietf:params:xml:ns:yang:ietf-igmp-mld-snooping
prefix:       ims
reference:    RFC XXXX
--------------------------------------------------------------------
```

7. References

7.1. Normative References

[dot1Qcp] IEEE, "Standard for Local and metropolitan area networks--Bridges and Bridged Networks--Amendment 30: YANG Data Model", IEEE Std 802.1Qcp-2018 (Revision of IEEE Std 802.1Q-2014), September 2018, <https://ieeexplore.ieee.org/servlet/opac?punumber=8467505>

[RFC1112] Deering, S., "Host extensions for IP multicasting", STD 5, RFC 1112, August 1989.

[RFC2236] W. Fenner, "Internet Group Management Protocol, Version 2", RFC 2236, November 1997.

[RFC2710] Deering, S., Fenner, W., and B. Haberman, "Multicast Listener Discovery (MLD) for IPv6", RFC 2710, October 1999.

[RFC3376] Cain, B., Deering, S., Kouvelas, I., Fenner, B., and A. Thyagarajan, "Internet Group Management Protocol, Version 3", RFC 3376, October 2002.

[RFC3688] Mealling, M., "The IETF XML Registry", RFC 3688, January 2004.

[RFC3810] Vida, R. and L. Costa, "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", RFC 3810, June 2004.

[RFC4286] B. Haberman and J. Martin, "Multicast Router Discovery", RFC 4286, December 2005.

[RFC4541] M. Christensen, K. Kimball, F. Solensky, "Considerations for Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD) Snooping Switches", RFC 4541, May 2006.

[RFC5790] H. Liu, W. Cao, H. Asaeda, "Lightweight Internet Group Management Protocol Version 3 (IGMPv3) and Multicast Listener Discovery Version 2 (MLDv2) Protocols", RFC 5790, February 2010.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[RFC6241] R. Enns, Ed., M. Bjorklund, Ed., J. Schoenwaelder, Ed., A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, June 2011.

[RFC6636] H. Asaeda, H. Liu, Q. Wu, "Tuning the Behavior of the Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD) for Routers in Mobile and Wireless Networks", RFC 6636, May 2012.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, July 2013.

[RFC7761] B. Fenner, M. Handley, H. Holbrook, I. Kouvelas, R. Parekh, Z. Zhang, L. Zheng, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", RFC 7761, March 2016.

[RFC7950] M. Bjorklund, Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, August 2016.

[RFC8040] A. Bierman, M. Bjorklund, K. Watsen, "RESTCONF Protocol", RFC 8040, January 2017.

[RFC8294] X. Liu, Y. Qu, A. Lindem, C. Hopps, L. Berger, "Common YANG Data Types for the Routing Area", RFC 8294, December 2017.

[RFC8340] M. Bjorklund, and L. Berger, Ed., "YANG Tree Diagrams", RFC 8340, March 2018.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", RFC 8341, March 2018.

[RFC8342] M. Bjorklund and J. Schoenwaelder, "Network Management Datastore Architecture (NMDA)", RFC 8342, March 2018.

[RFC8343] M. Bjorklund, "A YANG Data Model for Interface Management", RFC 8343, March 2018.

[RFC8349] L. Lhotka, A. Lindem, Y. Qu, "A YANG Data Model for Routing Management (NMDA Version)", RFC 8349, March 2018.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, August 2018.

[RFC8529] L. Berger, C. Hopps, A. Lindem, D. Bogdanovic, X. Liu, "YANG Data Model for Network Instances", RFC 8529, March 2019.

[draft-ietf-bess-l2vpn-yang] Shah, H., Brissette, P., Chen, I., Hussain, I., Wen, B., and K. Tiruveedhula, "YANG Data Model for MPLS-basedL2VPN", draft-ietf-bess-l2vpn-yang-10 (work in progress), July 2019.

7.2. Informative References

[RFC3916] X. Xiao, Ed., D. McPherson, Ed., P. Pate, Ed., "Requirements for Pseudo-Wire Emulation Edge-to-Edge (PWE3)", RFC 3916, September 2004.

[RFC7951] L. Lhotka, "JSON Encoding of Data Modeled with YANG", RFC 7951, August 2016.

[RFC8407] A. Bierman, "Guidelines for Authors and Reviewers of Documents Containing YANG Data Models", RFC 8407, October 2018.

[RFC8652] X. Liu, F. Guo, M. Sivakumar, P. McAllister, A. Peter, "A YANG Data Model for the Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD)", RFC 8652, November 2019.

Appendix A. Data Tree Example

A.1 Bridge service

This section contains an example for bridge service in the JSON encoding [RFC7951], containing both configuration and state data.

```
              +-----------+
              +  Source   +
              +-----+-----+
                    |
   -----------------+---------------------------
                    |eth1/1
                +---+---+
                +  R1   +
                +-+---+-+
           eth1/2 |    \ eth1/3
                  |     \
                  |      \
                  |       \
                  |        \
           eth2/1 |         \ eth3/1
              +---+---+   +--+---+
              +  R2   +   +  R3  +
              +---+---+   +--+---+
           eth2/2 |          | eth3/2
                  |          |
   ---------------+----------+------------------
                  |          |
                  |          |
         +--------+--+   +---+--------+
         + Receiver1 +   +  Receiver2 +
         +-----------+   +------------+
```

The configuration data for R1 in the above figure could be as follows:


```
{
"ietf-interfaces:interfaces":{
  "interface":[
   {
    "name":"eth1/1",
    "type":"iana-if-type:ethernetCsmacd"
   }
  ]
 },
 "ietf-routing:routing":{
  "control-plane-protocols":{
   "control-plane-protocol":[
    {
     "type":"ietf-igmp-mld-snooping:igmp-snooping",
     "name":"bis1",
     "ietf-igmp-mld-snooping:igmp-snooping-instance":{
```

```
       "l2-service-type":"ietf-igmp-mld-snooping:bridge",
       "enable":true
      }
     }
    ]
   }
  },
  "ieee802-dot1q-bridge:bridges":{
   "bridge":[
    {
     "name":"isp1",
     "address":"00-23-ef-a5-77-12",
     "bridge-type":"ieee802-dot1q-bridge:customer-vlan-bridge",
     "component":[
      {
       "name":"comp1",
       "type":"ieee802-dot1q-bridge:c-vlan-component",
       "bridge-vlan":{
        "vlan":[
         {
          "vid":101,
          "ietf-igmp-mld-snooping:igmp-snooping-instance":"bis1"
         }
        ]
       }
      }
     ]
    }
   ]
  }
}
```


The corresponding operational state data for R1 could be as follows:


```
{
 "ietf-interfaces:interfaces": {
  "interface": [
   {
    "name": "eth1/1",
    "type": "iana-if-type:ethernetCsmacd",
    "oper-status": "up",
    "statistics": {
     "discontinuity-time": "2018-05-23T12:34:56-05:00"
    }
   }
  ]
 },
 "ietf-routing:routing": {
  "control-plane-protocols": {
   "control-plane-protocol": [
    {
     "type": "ietf-igmp-mld-snooping:igmp-snooping",
```

```
     "name": "bis1",
     "ietf-igmp-mld-snooping:igmp-snooping-instance": {
      "l2-service-type": "ietf-igmp-mld-snooping:bridge",
      "enable": true
     }
    }
   ]
  }
 },
 "ieee802-dot1q-bridge:bridges": {
  "bridge": [
   {
    "name": "isp1",
    "address": "00-23-ef-a5-77-12",
    "bridge-type": "ieee802-dot1q-bridge:customer-vlan-bridge",
    "component": [
     {
      "name": "comp1",
      "type": "ieee802-dot1q-bridge:c-vlan-component",
      "bridge-vlan": {
       "vlan": [
        {
         "vid": 101,
         "ietf-igmp-mld-snooping:igmp-snooping-instance": "bis1"
        }
       ]
      }
     }
    ]
   }
  ]
 }
}
```


The following action is to clear all the entries whose group address is 225.1.1.1 for igmp-snooping-instance bis1.

```
POST /restconf/operations/ietf-routing:routing/control-plane-protocols/\
control-plane-protocol=ietf-igmp-mld-snooping:igmp-snooping,bis1/\
ietf-igmp-mld-snooping:igmp-snooping-instance/\
clear-igmp-snooping-groups HTTP/1.1
Host: example.com
Content-Type: application/yang-data+json
```


```
{
  "ietf-igmp-mld-snooping:input" : {
    "group": "225.1.1.1",
    "source": "*"
  }
}
```

A.2 L2VPN service

This section contains an example for L2VPN service in the JSON encoding [RFC7951], containing both configuration and state data.

```
              +------------+
              +  Source    +
              +-----+------+
                    |
   -----------------+----------------------------
                    |eth1/1
                +---+---+
                +  R1   +
                +-+---+-+
           eth1/2 |    \ eth1/3
                  |     \
                  |      \
                  |       \
                  |        \
           eth2/1 |         \ eth3/1
              +---+---+    +-+---+
              +  R2   +----+ R3  +
              +---+---+    +-+---+
           eth2/2 |          | eth3/2
                  |          |
   ---------------+----------+-------------------
                  |          |
                  |          |
         +--------+--+   +---+--------+
         + Receiver1 +   +  Receiver2 +
         +-----------+   +------------+
```

The configuration data for R1 in the above figure could be as follows:


```
{
"ietf-interfaces:interfaces":{
  "interface":[
   {
    "name":"eth1/1",
    "type":"iana-if-type:ethernetCsmacd"
   }
  ]
 },
 "ietf-pseudowires:pseudowires": {
  "pseudowire": [
   {
    "name": "pw2"
   },
   {
    "name": "pw3"
```

```
   }
  ]
 },
 "ietf-network-instance:network-instances": {
  "network-instance": [
   {
    "name": "vpls1",
    "ietf-igmp-mld-snooping:igmp-snooping-instance": "vis1",
    "ietf-l2vpn:type": "ietf-l2vpn:vpls-instance-type",
    "ietf-l2vpn:signaling-type": "ietf-l2vpn:ldp-signaling",
    "ietf-l2vpn:endpoint": [
     {
      "name": "acs",
      "ac": [
        {
         "name": "eth1/1"
        }
      ]
     },
     {
      "name": "pws",
      "pw": [
        {
         "name": "pw2"
        },
        {
         "name": "pw3"
        }
      ]
     }
    ]
   }
  ]
 },
 "ietf-routing:routing": {
  "control-plane-protocols": {
   "control-plane-protocol": [
    {
     "type": "ietf-igmp-mld-snooping:igmp-snooping",
     "name": "vis1",
     "ietf-igmp-mld-snooping:igmp-snooping-instance": {
      "l2-service-type": "ietf-igmp-mld-snooping:l2vpn",
      "enable": true
     }
    }
   ]
  }
 }
}
```

The corresponding operational state data for R1 could be as follows:


```
{
"ietf-interfaces:interfaces":{
  "interface":[
   {
    "name":"eth1/1",
    "type":"iana-if-type:ethernetCsmacd",
    "oper-status": "up",
    "statistics": {
     "discontinuity-time": "2018-05-23T12:34:56-05:00"
    }
   }
  ]
 },
 "ietf-pseudowires:pseudowires": {
  "pseudowire": [
   {
    "name": "pw2"
   },
   {
    "name": "pw3"
   }
  ]
 },
 "ietf-network-instance:network-instances": {
  "network-instance": [
   {
    "name": "vpls1",
    "ietf-igmp-mld-snooping:igmp-snooping-instance": "vis1",
    "ietf-l2vpn:type": "ietf-l2vpn:vpls-instance-type",
    "ietf-l2vpn:signaling-type": "ietf-l2vpn:ldp-signaling",
    "ietf-l2vpn:endpoint": [
     {
      "name": "acs",
      "ac": [
        {
         "name": "eth1/1"
        }
      ]
     },
     {
      "name": "pws",
      "pw": [
        {
         "name": "pw2"
        },
        {
         "name": "pw3"
        }
      ]
     }
```

```
    ]
   }
  ]
 },
 "ietf-routing:routing": {
  "control-plane-protocols": {
   "control-plane-protocol": [
    {
     "type": "ietf-igmp-mld-snooping:igmp-snooping",
     "name": "vis1",
     "ietf-igmp-mld-snooping:igmp-snooping-instance": {
      "l2-service-type": "ietf-igmp-mld-snooping:l2vpn",
      "enable": true
     }
    }
   ]
  }
 }
}
```

Authors' Addresses

Hongji Zhao
Ericsson (China) Communications Company Ltd.
Ericsson Tower, No. 5 Lize East Street,
Chaoyang District Beijing 100102, P.R. China

Email: hongji.zhao@ericsson.com

Xufeng Liu
Volta Networks
USA

EMail: xufeng.liu.ietf@gmail.com

Yisong Liu
China Mobile
China

Email: liuyisong@chinamobile.com

Anish Peter
Individual

EMail: anish.ietf@gmail.com

Mahesh Sivakumar
Juniper Networks
1133 Innovation Way
Sunnyvale, California
USA

EMail: sivakumar.mahesh@gmail.com

An Enhancement of PIM-SM

draft-meng-pim-sm-enhancement-01.txt

Abstract

This document specifies an enhanced version of PIM-SM which works without requiring whole network deployment.

Status of this Memo

Copyright and License Notice

Table of Contents

1. Introduction

PIM-SM is a multicast routing protocol that can use the underlying unicast routing information base or a separate multicast-capable routing information base. It builds unidirectional shared trees rooted at a Rendezvous Point (RP) per group, and it optionally creates shortest-path trees per source.

However, PIM-SM must be deployed contiguously in the whole network, because a joining router can not join into a tree if the upstream neighbor does not support PIM-SM.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Problem Description

PIM-SM protocol works generally as follows:

1)Multicast receivers express their interests in receiving traffic destined for multicast by using IGMP, MLD or other mechanisms.

2)One of a receiver’s local routers is elected as the Designated Router (DR) for that subnet. On receiving the receiver’s expression of interest, the DR then sends a PIM Join message towards the RP/Source for that multicast group. The Join message travels hop-by-hop towards the RP/Multicast source for the group, and in each router it passes through, multicast tree state for group G is instantiated. Eventually, the Join message either reaches the RP/Multicast source or reaches a router that already has Join state for that group.

3)In RFC 7761, all join/prune messages are multicast with TTL 1 to the ’ALL-PIM-ROUTERS’ group, a message receiver would compare Unicast Upstream Neighbor Address carried in the message with the address of itself, the message will be processed only if the two addresses are the same.

As long as there is one router does not support PIM in the path from a joining router to the target RP/Source, the router can not join into the RPT/SPT.

3. Compatible Scheme

This scheme is based on RFC 7761.

New unicast Join/Prune messages(and their process procedures) will be introduced in this scheme and they will coexist with old Join/Prune messages.

3.1. Sending Join/Prune Messages

PIM-SM routers send join messages to join into multicast groups, send prune messages to leave multicast groups.

If upstream neighbor is a PIM-SM neighbor, old join/prune messages should be sent by the joining/pruning router even if unicast join/prune messages are being received.

Otherwise, new unicast join/prune messages should be sent as below:

1)If an RPT is being joined/pruned, the destination address of the unicast join/prune message should be the RP address, the source address of the unicast join/prune message should be the address of the joining/pruning router.

2)If an SPT is being joined/pruned, the destination address of the unicast join/prune message should be the multicast source address, the source address of the unicast join/prune message should be the address of the joining/pruning router, and there is no Joined/Pruned Source Address field in the message.

3.2. Receiving Join Messages

Both old join messages and new unicast join messages could be received:

1)Old Join messages can only be received by PIM-SM neighbors of the sender, they should be processed according to RFC7761.

2)Unicast Join messages could be received by PIM routers(other than RP/Multicast Source) through ACL or similar means, they could also be received by the destination(RP/Multicast Source) of the messages, receivers should create tunnels from themselves to senders along with new states.

Join messages should be processed as below in detail:

```
join_msg_arrives(msg) {

    if (msg.dst == 224.0.0.13) {

        //The message should be processed according to RFC 7761
```

```
    } else {
        S = multicast_source(msg);
        state = S ? get_state(*, G) : get_state(S, G);
        if (!state) {
            state = S ? new_state(*, G) : new_state(S, G);
            if (msg.dst != self_addr) {
                if (upstream_neighbor_is_a_PIM-SM_neighbor) {
                    send_multicast_join_message;
                } else {
                    new_msg = msg;
                    new_msg.src = OIF(new_msg).addr;
                    send(new_msg);
                }
            }
        }
        add_IF_to_olist(state, create_tunnel(IIF(msg).addr,
msg.src));
    }
}
add_IF_to_olist(state, IF) {
    if (/*IF is in state’s olist*/){
        return;
    }
    add(state.olist, IF);
}
```

```
   IIF(msg) {

       return the input interface of msg;

   }

   OIF(msg) {

       return the output interface of msg;

   }
```

3.3. Receiving Prune Messages

Old Prune messages should be processed according to RFC7761.

New Prune messages would be intercepted by PIM routers or be received be RP/Source, they should be processed as below.

```
   prune_msg_arrives(msg) {

       if (msg.dst == 224.0.0.13) {

           //The message should be processed according to RFC 7761

       } else {

           S = multicast_source(msg);

           state = S ? get_state(*, G) : get_state(S, G);

           if (state) {

               IIF = tunnel(IIF(msg).addr, msg.src) ?
   tunnel(IIF(msg).addr, msg.src) : IIF(msg);

               delete_IF_from_olist(state, IIF);

               if (state.olist_num == 0) {

                   delete_state(state);

                   if (msg.dst != self_addr) {

                       if (upstream_neighbor_is_a_PIM-SM_neighbor) {

                           send_multicast_prune_message;
```

```
                    } else {
                        new_msg = msg;
                        new_msg.src = OIF(new_msg).addr;
                        send(new_msg);
                    }
                }
            }
        } else if (msg.dst != self_addr) {
            forward(msg);
        } else {
            //The prune message should be ignored
        }
    }
}
delete_IF_from_olist(state, IF) {
    if (/*IF is not in state’s olist*/){
        return;
    }
    delete(state.olist, IF);
}
IIF(msg) {
    return the input interface of msg;
}
OIF(msg) {
```

```
    return the output interface of msg;

}
```

4. Clean Slate Scheme

This scheme is a modification of RFC 7761:

1)Neighbor relationship between PIM routers will no longer be maintained.

2)Join/Prune messages(and their process procedures) in RFC 7761 will be replaced by Join/Prune messages introduced in this section.

4.1. Sending Join/Prune Messages

Join/Prune messages will no longer be multicast with TTL 1 to the 'ALL-PIM-ROUTERS' group, they will be unicast as below:

1)If an RPT is being joined/pruned, the destination address of the join/prune message should be the RP address, the source address of the join/prune message should be the address of the joining/pruning router.

2)If an SPT is being joined/pruned, the destination address of the join/prune message should be the multicast source address, the source address of the join/prune message should be the address of the joining/pruning router, and there is no Joined/Pruned Source Address field in the message.

4.2. Receiving Join Messages

Join messages could be received by PIM routers(other than RP/Multicast Source) through ACL or similar means, they could also be received by the destination(RP/Multicast Source) of the messages.

A receiver should create tunnel from itself to the sender along with new state only if it is the sender's neighbor which can be identified by TTL in IPv4 packet or Hop Limit in IPv6 packet.

Join messages would be intercepted by PIM routers or be received be RP/Source, they should be processed as below:

```
join_msg_arrives(msg) {

    S = multicast_source(msg);

    state = S ? get_state(*, G) : get_state(S, G);
```

```
    if (!state) {

        state = S ? new_state(*, G) : new_state(S, G);

        if (msg.dst != self_addr) {

            new_msg = msg;

            new_msg.src = OIF(new_msg).addr;

            new_msg.ttl = 255;

            send(new_msg);

        }

    }

    IIF = (msg.ttl == 255) ? IIF(msg) : create_tunnel(IIF(msg).addr,
msg.src);

    add_IF_to_olist(state, IIF);

}

add_IF_to_olist(state, IF) {

    if (/*IF is in state’s olist*/){

        return;

    }

    add(state.olist, IF);

}

IIF(msg) {

    return the input interface of msg;

}

OIF(msg) {

    return the output interface of msg;

}
```

4.3. Receiving Prune Messages

Prune messages would be intercepted by PIM routers or be received be RP/Source, they should be processed as below:

```
prune_msg_arrives(msg) {

    S = multicast_source(msg);

    state = S ? get_state(*, G) : get_state(S, G);

    if (state) {

        IIF = tunnel(IIF(msg).addr, msg.src) ? tunnel(IIF(msg).addr,
msg.src) : IIF(msg);

        delete_IF_from_olist(state, IIF);

        if (state.olist_num == 0) {

            delete_state(state);

            if (msg.dst != self_addr) {

                new_msg = msg;

                new_msg.src = OIF(new_msg).addr;

                send(new_msg);

            }

        }

    } else if (msg.dst != self_addr) {

        forward(msg);

    } else {

        //The prune message should be ignored

    }

}

delete_IF_from_olist(state, IF) {
```

```
        if (/*IF is not in state's olist*/){

            return;

        }

        delete(state.olist, IF);

    }

    IIF(msg) {

        return the input interface of msg;

    }

    OIF(msg) {

        return the output interface of msg;

    }
```

5. Packet Formats

There is only one modification about packet formats:

If an SPT is being joined/pruned, there will be no Joined/Pruned Source Address field in the joined/pruned message, and the Number of Joined Sources in the message is 1.

6. Security Considerations

To be perfected.

7. IANA Considerations

There is no IANA consideration in this specification.

8. References

8.1. Normative References

[KEYWORDS] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2119]   Bradner, S., "Key words for use in RFCs to Indicate

Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>..

[RFC7761] Fenner, B., Handley, M., Holbrook, H., Kouvelas, I., Parekh, R., Zhang, Z., and L. Zheng, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", STD 83, RFC 7761, DOI 10.17487/RFC7761, March 2016, <http://www.rfc-editor.org/info/rfc7761>.

Authors' Addresses

Rui Meng
Huawei Technologies Co., Ltd
Huawei Campus, 156 Beiqing Road, Hai-dian District
Beijing 100089
China

EMail: mengrui@huawei.com

PIM Working Group G. Mirsky
Internet-Draft ZTE Corp.
Updates: 7761 (if approved) J. Xiaoli
Intended status: Standards Track ZTE Corporation
Expires: August 19, 2018 February 15, 2018

Bidirectional Forwarding Detection (BFD) for Multi-point Networks and Protocol Independent Multicast - Sparse Mode (PIM-SM) Use Case
draft-mirsky-pim-bfd-p2mp-use-case-00

Abstract

This document discusses use of Bidirectional Forwarding Detection (BFD) for multi-point networks to provide nodes that participate in Protocol Independent Multicast - Sparse Mode (PIM-SM) over shared-media segment with sub-second convergence of the Designated Router and defines the extension to bootstrap point-to-multipoint BFD session. Optional extension to PIM-SM Hello, as defined in RFC 7761, also defined in this document.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

[RFC7761] is the current specification of the Protocol Independent Multicast - Sparse Mode (PIM-SM) for IPv4 and IPv6 networks. Confirming implementation of PIM-SM elects a Designated Router (DR) on each PIM-SM interface. When a group of PIM-SM nodes are connected to shared-media segment, e.g. Ethernet, the one elected as DR is to act on behalf of directly connected hosts in context of the PIM-SM protocol. Failure of the DR impacts quality of the multicast services it provides to directly connected hosts because the default failure detection interval for PIM-SM routers is 105 seconds. Introduction of Backup DR (BDR), proposed in [I-D.ietf-pim-dr-improvement] improves convergence time in the PIM-SM over shared-media segment but still depends on long failure detection interval.

Bidirectional Forwarding Detection (BFD) [RFC5880] had been originally defined to detect failure of point-to-point (p2p) paths - single-hop [RFC5881], multihop [RFC5883]. [I-D.ietf-bfd-multipoint] extends [RFC5880] for multipoint and multicast networks, which precisely characterizes deployment scenarios for PIM-SM over LAN segment. This document demonstrates how point-to-multipoint (p2mp) BFD can enable faster detection of PIM-SM DR and BDR failure and thus minimize multicast service disruption. The document also defines the extension to PIM-SM [RFC7761] to bootstrap a PIM-SM router to join in p2mp BFD session over shared-media link.

1.1. Conventions used in this document

1.1.1. Terminology

BFD: Bidirectional Forwarding Detection

BDR: Backup Designated Router

DR: Designated Router

p2mp: Pont-to-Multipoint

PIM-SM: Protocol Independent Multicast - Sparse Mode

1.1.2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Problem Statement

Several PIM-SM routers may be connected over shared-media link, e.g. Ethernet. [RFC7761] does not provide method for fast, e.g. sub-second, DR failure detection by other PIM-SM routers on the same Ethernet link. BFD already has many implementations based on HW that are capable to support multiple sub-second session concurrently. [Editor's note: monitoring of PIM-SM BDR liveliness will be addressed in the next update of the draft.]

3. Applicability of p2mp BFD

[I-D.ietf-bfd-multipoint] may provide the efficient and scalable solution for fast-converging environment that has head-tails relationships. Each such group presents itself as p2mp BFD session with its head being the root and other routers being tails of the p2mp BFD session. Figure 1 displays the new BFD Discriminator TLV [RFC7761] to bootstrap tail of the p2mp BFD session.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          OptionType           |         OptionLength          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        My  Discriminator                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: BFD Discriminator TLV to Bootstrap P2MP BFD session

where new fields are interpreted as:

OptionType is value (TBA1) assigned by IANA Section 4 that identifies the TLV as BFD Discriminator TLV;

OptionLength value is always 4

My Discriminator - My Discriminator value allocated by the root of the p2mp BFD session.

If PIM-SM routers, that support this specification, are configured to use p2mp BFD for faster convergence, then the DR MUST create BFD session MultipointHead, as defined in [I-D.ietf-bfd-multipoint]. PIM-SM DR MUST include BFD TLV in its PIM-Hello message. PIM-SM DR periodically transmits BFD control packets. Source IP address of the BFD control packet MUST be the same as the source IP address of the PIM-Hello with BFD TLV messages being transmitted by the DR. The values of My Discriminator in the BFD control packet and My Discriminator field of the BFD TLV in PIM-Hello, transmitted by the PIM-SM DR, MUST be the same. When non-DR PIM-SM router receives PIM-Hello packet from DR with BFD TLV it MAY create p2mp BFD session as MultipointTail, as defined in [I-D.ietf-bfd-multipoint], and demultiplex p2mp BFD test session based on DR source IP address the My Discriminator value value it learned from BFD Discriminator TLV. If DR ceased to include BFD TLV in its PIM-Hello message, other PIM-SM nodes MUST close corresponding MultipointTail BFD session.

3.1. Multipoint BFD Encapsulation

The MultipointHead of p2mp BFD session when transmitting BFD control packet:

MUST set TTL value to 1;

SHOULD use group address ALL-PIM-ROUTERS (’224.0.0.13’ for IPv4 and ’ff02::d’ for IPv6) as destination IP address

MAY use network broadcast address for IPv4 or link-local all nodes multicast group for IPv6 as destination IP address;

MUST set destination UDP port value to 3784 when transmitting BFD control packets, as defined in [I-D.ietf-bfd-multipoint].

4. IANA Considerations

IANA is requested to allocate new OptionType value from PIM Hello Options registry according to:

Value Name	Length Number	Name Protocol	Reference
TBA	4	BFD Discriminator	This document

Table 1: BFD Discriminator option type

5. Security Considerations

Security considerations discussed in [RFC7761], [RFC5880], and [I-D.ietf-bfd-multipoint], apply to this document.

6. Acknowledgements

TBD

7. Normative References

[I-D.ietf-bfd-multipoint]
Katz, D., Ward, D., Networks, J., and G. Mirsky, "BFD for Multipoint Networks", draft-ietf-bfd-multipoint-13 (work in progress), January 2018.

[I-D.ietf-pim-dr-improvement]
Zhang, Z., hu, f., Xu, B., and m. mishra, "PIM DR Improvement", draft-ietf-pim-dr-improvement-04 (work in progress), December 2017.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <https://www.rfc-editor.org/info/rfc5880>.

[RFC5881] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD) for IPv4 and IPv6 (Single Hop)", RFC 5881, DOI 10.17487/RFC5881, June 2010, <https://www.rfc-editor.org/info/rfc5881>.

[RFC5883] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD) for Multihop Paths", RFC 5883, DOI 10.17487/RFC5883, June 2010, <https://www.rfc-editor.org/info/rfc5883>.

[RFC7761] Fenner, B., Handley, M., Holbrook, H., Kouvelas, I., Parekh, R., Zhang, Z., and L. Zheng, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", STD 83, RFC 7761, DOI 10.17487/RFC7761, March 2016, <https://www.rfc-editor.org/info/rfc7761>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

Authors' Addresses

Greg Mirsky
ZTE Corp.

Email: gregimirsky@gmail.com

Ji Xiaoli
ZTE Corporation
No.50 Software Avenue, Yuhuatai District
Nanjing
China

Email: ji.xiaoli@zte.com.cn

PIM Working Group G. Mirsky
Internet-Draft ZTE Corp.
Updates: 7761 (if approved) J. Xiaoli
Intended status: Standards Track ZTE Corporation
Expires: December 29, 2018 June 27, 2018

Bidirectional Forwarding Detection (BFD) for Multi-point Networks and Protocol Independent Multicast - Sparse Mode (PIM-SM) Use Case
draft-mirsky-pim-bfd-p2mp-use-case-02

Abstract

This document discusses the use of Bidirectional Forwarding Detection (BFD) for multi-point networks to provide nodes that participate in Protocol Independent Multicast - Sparse Mode (PIM-SM) with the sub-second convergence. Optional extension to PIM-SM Hello, as specified in RFC 7761, to bootstrap point-to-multipoint BFD session. also defined in this document.

Table of Contents

1. Introduction

Faster convergence in the control plane, in general, is beneficial and allows minimizing periods of traffic blackholing, transient routing loops and other scenarios that may negatively affect service data flow. That equally applies to unicast and multicast routing protocols.

[RFC7761] is the current specification of the Protocol Independent Multicast - Sparse Mode (PIM-SM) for IPv4 and IPv6 networks. Confirming implementation of PIM-SM elects a Designated Router (DR) on each PIM-SM interface. When a group of PIM-SM nodes is connected to shared-media segment, e.g. Ethernet, the one elected as DR is to act on behalf of directly connected hosts in context of the PIM-SM protocol. Failure of the DR impacts the quality of the multicast services it provides to directly connected hosts because the default failure detection interval for PIM-SM routers is 105 seconds. Introduction of Backup DR (BDR), proposed in [I-D.ietf-pim-dr-improvement] improves convergence time in the PIM-SM over shared-media segment but still depends on long failure detection interval.

Bidirectional Forwarding Detection (BFD) [RFC5880] had been originally defined to detect failure of point-to-point (p2p) paths - single-hop [RFC5881], multihop [RFC5883]. [I-D.ietf-bfd-multipoint] extends the BFD base specification [RFC5880] for multipoint and multicast networks, which precisely characterizes deployment scenarios for PIM-SM over LAN segment. This document demonstrates how point-to-multipoint (p2mp) BFD can enable faster detection of

PIM-SM router ailure and thus minimize multicast service disruption. The document also defines the extension to PIM-SM [RFC7761] to bootstrap a PIM-SM router to join in p2mp BFD session over shared-media link.

1.1. Conventions used in this document

1.1.1. Terminology

BFD: Bidirectional Forwarding Detection

BDR: Backup Designated Router

DR: Designated Router

p2mp: Pont-to-Multipoint

PIM-SM: Protocol Independent Multicast - Sparse Mode

1.1.2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Problem Statement

[RFC7761] does not provide a method for fast, e.g. sub-second, failure detection of a neighbor PIM-SM router. BFD already has many implementations based on HW that are capable to support multiple sub-second session concurrently.

3. Applicability of p2mp BFD

[I-D.ietf-bfd-multipoint] may provide the efficient and scalable solution for the fast-converging environment that has head-tails relationships. Each such group presents itself as p2mp BFD session with its head being the root and other routers being tails of the p2mp BFD session. Figure 1 displays the new BFD Discriminator TLV [RFC7761] to bootstrap tail of the p2mp BFD session.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          OptionType           |         OptionLength          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        My  Discriminator                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: BFD Discriminator TLV to Bootstrap P2MP BFD session

where new fields are interpreted as:

OptionType is a value (TBA1) assigned by IANA Section 4 that identifies the TLV as BFD Discriminator TLV;

OptionLength value is always 4

My Discriminator - My Discriminator value allocated by the root of the p2mp BFD session.

If PIM-SM routers, that support this specification, are configured to use p2mp BFD for faster convergence, then the router to be monitored, referred to as 'head', MUST create BFD session MultipointHead, as defined in [I-D.ietf-bfd-multipoint]. The head MUST include BFD TLV in its PIM-Hello message and periodically transmit BFD control packets. Source IP address of the BFD control packet MUST be the same as the source IP address of the PIM-Hello with BFD TLV messages being transmitted by the head. The values of My Discriminator in the BFD control packet and My Discriminator field of the BFD TLV in PIM-Hello, transmitted by the head MUST be the same. When a PIM-SM router configured to monitor the head, referred to as 'tail', via p2mp BFD receives PIM-Hello packet with BFD TLV it MAY create p2mp BFD session as MultipointTail, as defined in [I-D.ietf-bfd-multipoint], and demultiplex p2mp BFD test session based on head's source IP address the My Discriminator value it learned from BFD Discriminator TLV. If the head ceased to include BFD TLV in its PIM-Hello message, tails MUST close the corresponding MultipointTail BFD session. If the tail detects MultipointHead failure it MUST remove the neighbor. If the failed head node was PIM-SM DR or BDR the tail MAY start DR Election process as specified in Section 4.3.2 [RFC7761] or in Section 4.1 [I-D.ietf-pim-dr-improvement] respectively.

3.1. Multipoint BFD Encapsulation

The MultipointHead of p2mp BFD session when transmitting BFD control packet:

MUST set TTL value to 1;

SHOULD use group address ALL-PIM-ROUTERS ('224.0.0.13' for IPv4 and 'ff02::d' for IPv6) as destination IP address

MAY use network broadcast address for IPv4 or link-local all nodes multicast group for IPv6 as the destination IP address;

MUST set destination UDP port value to 3784 when transmitting BFD control packets, as defined in [I-D.ietf-bfd-multipoint].

4. IANA Considerations

IANA is requested to allocate a new OptionType value from PIM Hello Options registry according to:

Value Name	Length Number	Name Protocol	Reference
TBA	4	BFD Discriminator	This document

Table 1: BFD Discriminator option type

5. Security Considerations

Security considerations discussed in [RFC7761], [RFC5880], and [I-D.ietf-bfd-multipoint], apply to this document.

6. Acknowledgments

Authors cannot say enough to express their appreciation of comments and suggestions we received from Stig Venaas.

7. Normative References

[I-D.ietf-bfd-multipoint]
Katz, D., Ward, D., Networks, J., and G. Mirsky, "BFD for Multipoint Networks", draft-ietf-bfd-multipoint-18 (work in progress), June 2018.

[I-D.ietf-pim-dr-improvement]
Zhang, Z., hu, f., Xu, B., and m. mishra, "PIM DR Improvement", draft-ietf-pim-dr-improvement-04 (work in progress), December 2017.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <https://www.rfc-editor.org/info/rfc5880>.

[RFC5881] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD) for IPv4 and IPv6 (Single Hop)", RFC 5881, DOI 10.17487/RFC5881, June 2010, <https://www.rfc-editor.org/info/rfc5881>.

[RFC5883] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD) for Multihop Paths", RFC 5883, DOI 10.17487/RFC5883, June 2010, <https://www.rfc-editor.org/info/rfc5883>.

[RFC7761] Fenner, B., Handley, M., Holbrook, H., Kouvelas, I., Parekh, R., Zhang, Z., and L. Zheng, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", STD 83, RFC 7761, DOI 10.17487/RFC7761, March 2016, <https://www.rfc-editor.org/info/rfc7761>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

Authors' Addresses

Greg Mirsky
ZTE Corp.

Email: gregimirsky@gmail.com

Ji Xiaoli
ZTE Corporation
No.50 Software Avenue, Yuhuatai District
Nanjing
China

Email: ji.xiaoli@zte.com.cn

Network Working Group V. Kamath
Internet-Draft R. Chokkanathapuram Sundaram
Intended status: Standards Track Cisco Systems, Inc.
Expires: August 31, 2018 February 27, 2018

PIM NULL Register packing
draft-ramki-pim-null-register-packing-00

Abstract

In PIM-SM networks PIM registers are sent from the first hop router to the RP (Rendezvous Point) to signal the presence of Multicast source in the network.There are periodic PIM Null registers sent from first hop router to the RP to keep the state alive at the RP as long as the source is active. The PIM null register packet carry information about a single Multicast source and group. This document defines a standard to send multiple Multicast source and group information in a single pim null register packet and the interoperability between the PIM routers which do not understand the packet format with multiple Multicast source and group details.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

PIM Null registers are sent by First hop routers periodically for Multicast streams to keep the states active on the RP as long as the Multicast source is alive. As the number of multicast sources increase, the number of PIM null register packets that are sent increases at a given time. This results in more PIM packet processing at RP and FHR. The control plane policing(COPP), monitors the packets that gets processed by the control plane. Due to the high rate at which NULL registers are received at the RP, this can lead to COPP drops of Multicast PIM null register packets. This draft proposes a method to efficiently pack multiple PIM null registers and register stop into a single message as these packets anyway don't contain data. The draft also proposes interoperability with the routers that do not understand the new packet format.

1.1. Conventions Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

1.2. Terminology

RP: Rendezvous Point

RPF: Reverse Path Forwarding

SPT: Shortest Path Tree

FHR: First Hop Router, directly connected to the source

LHR: Last Hop Router, directly connected to the receiver

2. PIM Register Stop format with capability option

A router (FHR) can decide to pack multiple NULL registers based on the capability received from the RP as part of Register Stop. This ensures compatibility with routers that don't support processing of the new format. The capability information can be indicated by the RP via the PIM register stop message sent to the FHR. Thus a FHR will switch to the new format only when it learns RP is capable of handling the packed null register messages. Conversely, a FHR that doesn't support the new format can continue generating the PIM NULL register the usual way since they don't check for the capability information present in the Register stop message. To exchange the capability information in the Register Stop message, the "reserved" field can be used to indicate this capability in those register stop messages. One bit of the reserved field is used to indicate the "packing" capability (P bit). The rest of the bits in the "Reserved" field will be retained for future use.

Figure 2: PIM Register Stop packet with capability option

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|PIM Ver| Type  |P|  Reserved   |                   Checksum    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             Group Address (Encoded-Group format)              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Source Address (Encoded-Unicast format)            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

PIM Version, Reserved, Type, Checksum, Group Address, Source Address
    Same as RFC 7761 (Section 4.9.4)

P   Capability bit used to indicate support for Packed NULL Register

3. New PIM Null register format

PIM null-register packet format is enhanced to include the count of the number of null-register records and pack multiple null-register records in the same packet. Currently the data part in the NULL register packet is a dummy IPv4 header which carries the source and group information and the other fields are unused. To indicate that the null register is in a new format the "Type" field in the PIM register packet format is used. To indicate the number of null register records a new field "record count" is introduced which can hold 8 bit value (max 255 records can be packed) which can be based on MTU. Even though null registers are supposed to be sent exactly every 60s, its fine to send a null register earlier, so as to merge the registers. When one register is sent, multiple registers can be packed together which are close enough in time.

Figure 1: New PIM NULL Register packet format

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |PIM Ver| Type  |    Reserved   |           Checksum            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    | Record count  |              Reserved2                        |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |     Group Address[1]  (Encoded-Group format)                  |
    |     Source Address[1]  (Encoded-Unicast format)               |
    .                                                               .
    .                                                               .
    .                                                               .
    .                                                               .
    .     Group Address[N]                                          .
    |     Source Address[N]                                         |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

PIM Version, Reserved, Checksum
    Same as RFC 7761 (Section 4.9.3)

Type
    The new packed NULL Register type value TBD

Record count
    The count of the number of packed NULL register records.
    A record consists of Group and Source Address

Group Address
    IP address of the Multicast Group

Source Address
    IP Address of the Multicast Source

4. New Packed PIM Register Stop format

The PIM register stop can be optimized to include multiple multicast group and source information. The Record count can indicate the number of S,G records that are packed and the Type value is used to indicate the new format.

Figure 3: New PIM Packed Register Stop packet formats

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|PIM Ver| Type  |  Reserved     |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Record count  |          Reserved2                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Group Address[1]  (Encoded-Group format)                  |
|     Source Address[1]  (Encoded-Unicast format)               |
.                                                               .
.                                                               .
.                                                               .
.                                                               .
.     Group Address[N]                                          .
|     Source Address[N]                                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

PIM Version, Reserved, Checksum
    Same as RFC 7761 (Section 4.9.3)

Type
    The new packed Register Stop type value TBD

Record count
    The count of the number of packed register stop records.
    A record consists of Group and Source Address

Group Address
    IP address of the Multicast Group

Source Address
    IP Address of the Multicast Source

5. Protocol operation

The following combinations exist -
FHR and RP both support the new PIM Register formats -

a. FHR sends the PIM register towards the RP when a new source is detected
b. RP sends a modified register stop towards the FHR that includes capabili
ty
information by setting the P bit (Figure 2)
c. Based on the receipt of the modified Register Stop, FHR will start packi
ng
of NULL registers using the new packed register format (Figure 1)
d. RP processes the NULL registers and can generate Register Stop messages
by
packing multiple S,Gs towards the same FHR (Figure 3)

FHR supports but RP doesn’t support new PIM Register formats-

a. FHR sends the PIM register towards the RP
b. RP sends a normal register stop without any capability information
c. FHR then sends NULL registers in the old format

RP supports but FHR doesn’t support the new PIM Register formats-

a. FHR sends the PIM register towards the RP
b. RP sends a modified register stop towards the FHR that includes capability information
c. Since FHR doesn’t support the new format, it sends NULL registers in the old format

6. IANA Considerations

This document requires the assignment of 2 new PIM message types for the packed pim register and pim register stop.

7. Acknowledgments

The authors would like to thank Stig Venaas and Umesh Dudani for contributing to the original idea and also their very helpful comments on the draft.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC7761] Fenner, B., Handley, M., Holbrook, H., Kouvelas, I., Parekh, R., Zhang, Z., and L. Zheng, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", STD 83, RFC 7761, DOI 10.17487/RFC7761, March 2016, <https://www.rfc-editor.org/info/rfc7761>.

8.2. Informative References

[RFC3973] Adams, A., Nicholas, J., and W. Siadak, "Protocol Independent Multicast - Dense Mode (PIM-DM): Protocol Specification (Revised)", RFC 3973, DOI 10.17487/RFC3973, January 2005, <https://www.rfc-editor.org/info/rfc3973>.

Authors' Addresses

Vikas Ramesh Kamath
Cisco Systems, Inc.
Tasman Drive
San Jose CA 95134
USA

Email: vikkamat@cisco.com

Ramakrishnan Cokkanathapuram Sundaram
Cisco Systems, Inc.
Tasman Drive
San Jose CA 95134
USA

Email: ramaksun@cisco.com

Network Working Group V. Kamath
Internet-Draft VMware
Intended status: Standards Track R. Chokkanathapuram Sundaram
Expires: October 14, 2019 R. Banthia
Cisco Systems, Inc.
April 12, 2019

PIM Null register packing
draft-ramki-pim-null-register-packing-03

Abstract

In PIM-SM networks PIM registers are sent from the first hop router to the RP (Rendezvous Point) to signal the presence of Multicast source in the network. There are periodic PIM Null registers sent from first hop router to the RP to keep the state alive at the RP as long as the source is active. The PIM Null register packet carries information about a single Multicast source and group. This document defines a standard to send multiple Multicast source and group information in a single pim Null register packet and the interoperability between the PIM routers which do not understand the packet format with multiple Multicast source and group details.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

PIM Null registers are sent by First hop routers periodically for Multicast streams to keep the states active on the RP as long as the Multicast source is alive. As the number of multicast sources increases, the number of PIM Null register packets that are sent increases at a given time. This results in more PIM packet processing at RP and FHR. The control plane policing (COPP), monitors the packets that gets processed by the control plane. Due to the high rate at which Null registers are received at the RP, this can lead to COPP drops of Multicast PIM Null register packets. This draft proposes a method to efficiently pack multiple PIM Null registers and register stop into a single message as these packets anyway don't contain data. The draft also proposes interoperability with the routers that do not understand the new packet format.

1.1. Conventions Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

1.2. Terminology

RP: Rendezvous Point

RPF: Reverse Path Forwarding

SPT: Shortest Path Tree

FHR: First Hop Router, directly connected to the source

LHR: Last Hop Router, directly connected to the receiver

2. PIM Register Stop format with capability option

A router (FHR) can decide to pack multiple Null registers based on the capability received from the RP as part of Register Stop. This ensures compatibility with routers that don't support processing of the new format. The capability information can be indicated by the RP via the PIM register stop message sent to the FHR. Thus a FHR will switch to the new format only when it learns RP is capable of handling the packed Null register messages. Conversely, a FHR that doesn't support the new format can continue generating the PIM Null register the current way. To exchange the capability information in the Register Stop message, the "reserved" field can be used to indicate this capability in those register stop messages. One bit of the reserved field is used to indicate the "packing" capability (P bit). The rest of the bits in the "Reserved" field will be retained for future use.

Figure 1: PIM Register Stop message with capability option

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|PIM Ver| Type  |P|  Reserved   |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             Group Address (Encoded-Group format)              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Source Address (Encoded-Unicast format)            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

PIM Version, Reserved, Type, Checksum, Group Address, Source Address
    Same as RFC 7761 (Section 4.9.4)

P   Capability bit used to indicate support for Packed Null Register

3. New PIM Null register message

New PIM Null register message format includes a count to indicate the number of Null register records in the message.

Figure 2: New PIM Null Register message format

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |PIM Ver| Type  |SubType| Rsvd  |           Checksum            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |   count       |              Reserved2                        |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |      Group Address[1]  (Encoded-Group format)                 |
    |      Source Address[1]  (Encoded-Unicast format)              |
    .                                                               .
    .                                                               .
    .                                                               .
    .                                                               .
    .      Group Address[N]                                         .
    |      Source Address[N]                                        |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

PIM Version, Reserved, Checksum
    Same as RFC 7761 (Section 4.9.3)

Type, SubType
    The new packed Null Register Type and SubType values TBD

count
    The count of the number of packed Null register records.
    A record consists of Group and Source Address

Group Address
    IP address of the Multicast Group

Source Address
    IP Address of the Multicast Source

4. New PIM Register Stop message format

The new PIM register stop is message includes a count to indicate the number of records that are present in the message.

Figure 3: New PIM Register Stop message format

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |PIM Ver| Type  |SubType| Rsvd  |           Checksum            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     count     |          Reserved2                            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Group Address[1]  (Encoded-Group format)                  |
   |     Source Address[1]  (Encoded-Unicast format)               |
   .                                                               .
   .                                                               .
   .                                                               .
   .                                                               .
   .     Group Address[N]                                          .
   |     Source Address[N]                                         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

PIM Version, Reserved, Checksum
    Same as RFC 7761 (Section 4.9.3)

Type
    The new Register Stop Type and SubType values TBD

Record count
    The count of the number of packed register stop records.
    A record consists of Group and Source Address

Group Address
    IP address of the Multicast Group

Source Address
    IP Address of the Multicast Source

5. Protocol operation

The following combinations exist -
FHR and RP both support the new PIM Register formats -

a. FHR sends the PIM register towards the RP when a new source is detected
b. RP sends a modified register stop towards the FHR that includes capability
   information by setting the P bit (Figure 2)
c. Based on the receipt of new Register Stop, FHR will
   start packing of Null registers using the new packed register format (Figure 1)
d. RP processes the new Null register message and can generate new register Stop messages by packing multiple S,Gs towards the same FHR (Figure 3)

FHR supports but RP doesn’t support new PIM Register formats-

a. FHR sends the PIM register towards the RP
b. RP sends a normal register stop without any capability information
c. FHR then sends Null registers in the old format

RP supports but FHR doesn’t support the new PIM Register formats-

a. FHR sends the PIM register towards the RP
b. RP sends a modified register stop towards the FHR that includes capability information
c. Since FHR doesn’t support the new format, it sends Null registers in the old format

6. PIM Anycast RP considerations

The new PIM register format should be enabled only if its supported by all PIM anycast RP members in the RP set for the RP address.

7. IANA Considerations

This document requires the assignment of 2 new PIM message types for the packed pim register and pim register stop.

8. Acknowledgments

The authors would like to thank Stig Venaas and Umesh Dudani for contributing to the original idea and also their very helpful comments on the draft.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC7761] Fenner, B., Handley, M., Holbrook, H., Kouvelas, I., Parekh, R., Zhang, Z., and L. Zheng, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", STD 83, RFC 7761, DOI 10.17487/RFC7761, March 2016, <https://www.rfc-editor.org/info/rfc7761>.

9.2. Informative References

[RFC3973] Adams, A., Nicholas, J., and W. Siadak, "Protocol Independent Multicast - Dense Mode (PIM-DM): Protocol Specification (Revised)", RFC 3973, DOI 10.17487/RFC3973, January 2005, <https://www.rfc-editor.org/info/rfc3973>.

Authors' Addresses

Vikas Ramesh Kamath
VMware
3401 Hillview Ave
Palo Alto CA 94304
USA

Email: vkamath@vmware.com

Ramakrishnan Chokkanathapuram Sundaram
Cisco Systems, Inc.
Tasman Drive
San Jose CA 95134
USA

Email: ramaksun@cisco.com

Raunak Banthia
Cisco Systems, Inc.
Tasman Drive
San Jose CA 95134
USA

Email: rbanthia@cisco.com

Network Working Group S. Venaas
Internet-Draft Cisco Systems, Inc.
Updates: 3973, 5015, 6754, 7761, ietf-pim-source-discovery-bsr (if approved) A. Retana
Huawei R&D USA
March 1, 2018
Intended status: Standards Track
Expires: September 2, 2018

PIM reserved bits and type space extension
draft-venaas-pim-reserved-bits-00

Abstract

The currently defined PIM version 2 messages share a common message header format. The common header definition contains eight reserved bits. This document specifies how these bits may be used by individual message types, and creates a registry containing the per message type usage. This document also extends the PIM type space by defining a new message type where four of the flag bits are used as an extended type range.

This document Updates RFC7761 and RFC3973 by defining the use of the currently Reserved field in the PIM common header. This document further updates RFC7761 and RFC3973, along with RFC5015, RFC6754 and I-D.ietf-pim-source-discovery-bsr, by specifying the use of the currently Reserved bits for each PIM message.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The currently defined PIM version 2 messages share a common message header format defined in the PIM Sparse Mode [RFC7761] and Dense Mode [RFC3973] specifications. The common header definition contains eight reserved bits. The message types defined in these documents all use this common header. However, several messages already make use of one or more bits, including the Bootstrap [RFC5059], DF-Election [RFC5015], and PIM Flooding Mechanism (PFM) [I-D.ietf-pim-source-discovery-bsr] messages. There is no document formally specifying that these bits are to be used per message type.

This document refers to the bits specified as Reserved in the common PIM header [RFC7761] [RFC3973] as PIM message type flag bits, or simply flag bits, and it specifies that they are to be separately used on a per message type basis. It creates a registry containing the the per message type usage. For a particular message type, the usage of the flag bits can be defined in the document defining the message type, or a new document that updates that document.

The PIM message types as defined in the PIM Sparse Mode [RFC7761] and Dense Mode [RFC3973] specifications are in the range from 0 to 15. That type space is almost exhausted. Message type 15 was reserved by [RFC6166] for type space extension. In Section 5, this document specifies the use of the flag bits for Type 15 in order to extend the PIM type space. The registration procedure for the extended type space is the same as for the existing type space, and the existing PIM message type registry is updated to include the extended type space.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. PIM header common format

The common PIM header is defined in section 4.9 of [RFC7761] and section 4.7.1 of [RFC3973]. This document updates the definition of the Reserved field and refers to that field as PIM message type flag bits, or simply flag bits. The new common header format is as below.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|PIM Ver| Type  |   Flags Bits  |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Flags Bits field is defined in Section 4. All other fields remain unchanged.

4. Flag Bit definitions

Unless otherwise specified, all the flag bits for each PIM type are Reserved [RFC8126]. They MUST be set to zero on transmission, and they MUST be ignored upon receipt. The specification of a new PIM type, MUST indicate whether the bits should be treated differently. Currently for the message types 0 (Hello), 1 (Register), 2 (Register Stop), 3 (Join/Prune), 5 (Assert), 6 (Graft), 7 (Graft-Ack), 8

(Candidate RP Advertisement), 9 (State Refresh) and 11 (ECMP Redirect), all flag bits are Reserved.

When defining flag bits it is helpful to have a well defined way of referring to a particular bit. The most significant of the flag bits, the bit immediately following the type field is referred to as bit 7. The least significant, the bit right in front of the checksum field is referred to as bit 0. This is shown in the diagram below.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|PIM Ver| Type  |7 6 5 4 3 2 1 0|           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

4.1. Flag Bits for Type 4 (Bootstrap)

PIM message type 4 (Bootstrap) [RFC5059] defines flag bit 7 as No-Forward. The usage of the bit is defined in that document. The remaining flag bits are Reserved.

4.2. Flag Bits for Type 10 (DF Election)

PIM message type 10 (DF Election) [RFC5015] specifies that the four most significant flag bits (bits 4-7) are to be used as a sub-type. The remaining flag bits are currently Reserved.

4.3. Flag Bits for Type 12 (PFM)

PIM message type 12 (PFM) [I-D.ietf-pim-source-discovery-bsr] defines flag bit 7 as No-Forward. The usage of the bit is defined in that document. The remaining flag bits are Reserved.

4.4. Flag Bits for Type 15 (Type Space Extension)

This type and the flag bit usage is defined in Section 5.

5. PIM Type Space Extension

The type space defined by the existing PIM specifications is almost exhausted. This document defines type 15 (Type Space Extension) allowing for 16 additional types by using the four most significant flag bits (bits 4-7) as a new field to store the extended type. These types are referred to as types 15.0 to 15.15 where the last number denotes the value stored in the new field. The remaining four flag bits (bits 0-3) are Reserved to be used by each extended type. The specification of a new PIM extended type MUST indicate whether the bits should be treated differently. The common header for types

15.0 to 15.15 is shown in the diagram below. The extended type field "Rsvd" denotes the value after the dot.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|PIM Ver|Type 15|SubType| Rsvd  |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6. Security Considerations

This document clarifies the use of the flag bits in the common PIM header and it extends the PIM type space. As such, there is no impact on security or changes to the considerations in [RFC7761] and [RFC3973].

7. IANA considerations

This document updates the PIM Message Types registry and also creates a PIM Message Type Flag Bits registry that shows which flag bits are defined for use by each of the PIM message types.

The following changes should be made to the existing PIM Message Types registry. For types 4 (Bootstrap) and 8 (Candidate RP Advertisement) a reference to RFC5059 should be added. For type 15 (Reserved), the name should be changed to "Type Space Extension", and reference this document. In addition, there should be one new row at the bottom where it should say "15.0-15.15 Unassigned".

A new registry called "PIM Message Type Flag Bits" should be created in the pim-paremeters section with registration procedure "IETF Review" as defined in [RFC8126] with this document as a reference. The initial content of the registry should be as below.

Type	bit(s)	Name	Reference
0	0-7	Reserved	[RFC3973][RFC7761]
1	0-7	Reserved	[RFC3973][RFC7761]
2	0-7	Reserved	[RFC3973][RFC7761]
3	0-7	Reserved	[RFC3973][RFC7761]
4	0-6	Reserved	[RFC3973][RFC7761]
4	7	No-Forward	[RFC5059]
5	0-7	Reserved	[RFC3973][RFC7761]
6	0-7	Reserved	[RFC3973][RFC7761]
7	0-7	Reserved	[RFC3973][RFC7761]
8	0-7	Reserved	[RFC3973][RFC7761]
9	0-7	Reserved	[RFC3973][RFC7761]
10	0-3	Reserved	[RFC3973][RFC7761]
10	4-7	Sub-type	[RFC5015]
11	0-7	Reserved	[RFC6754]
12	0-6	Reserved	[RFC3973][RFC7761]
12	7	No-Forward	[draft-ietf-pim-source-discovery-bsr]
13	0-7	Reserved	[RFC3973][RFC7761]
14	0-7	Reserved	[RFC3973][RFC7761]
15	4-7	Extended type	[this document]
15.0-15.15	0-3	Reserved	[this document]

8. References

8.1. Normative References

[I-D.ietf-pim-source-discovery-bsr]
Wijnands, I., Venaas, S., Brig, M., and A. Jonasson, "PIM Flooding Mechanism and Source Discovery", draft-ietf-pim-source-discovery-bsr-12 (work in progress), January 2018.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3973] Adams, A., Nicholas, J., and W. Siadak, "Protocol Independent Multicast - Dense Mode (PIM-DM): Protocol Specification (Revised)", RFC 3973, DOI 10.17487/RFC3973, January 2005, <https://www.rfc-editor.org/info/rfc3973>.

[RFC5015] Handley, M., Kouvelas, I., Speakman, T., and L. Vicisano, "Bidirectional Protocol Independent Multicast (BIDIR-PIM)", RFC 5015, DOI 10.17487/RFC5015, October 2007, <https://www.rfc-editor.org/info/rfc5015>.

[RFC5059] Bhaskar, N., Gall, A., Lingard, J., and S. Venaas, "Bootstrap Router (BSR) Mechanism for Protocol Independent Multicast (PIM)", RFC 5059, DOI 10.17487/RFC5059, January 2008, <https://www.rfc-editor.org/info/rfc5059>.

[RFC6754] Cai, Y., Wei, L., Ou, H., Arya, V., and S. Jethwani, "Protocol Independent Multicast Equal-Cost Multipath (ECMP) Redirect", RFC 6754, DOI 10.17487/RFC6754, October 2012, <https://www.rfc-editor.org/info/rfc6754>.

[RFC7761] Fenner, B., Handley, M., Holbrook, H., Kouvelas, I., Parekh, R., Zhang, Z., and L. Zheng, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", STD 83, RFC 7761, DOI 10.17487/RFC7761, March 2016, <https://www.rfc-editor.org/info/rfc7761>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

8.2. Informative References

[RFC6166] Venaas, S., "A Registry for PIM Message Types", RFC 6166, DOI 10.17487/RFC6166, April 2011, <https://www.rfc-editor.org/info/rfc6166>.

Authors’ Addresses

Stig Venaas
Cisco Systems, Inc.
Tasman Drive
San Jose CA 95134
USA

Email: stig@cisco.com

Alvaro Retana
Huawei R&D USA
2330 Central Expressway
Santa Clara CA 95050
USA

Email: alvaro.retana@huawei.com

Network Working Group
Internet-Draft
Updates: 3973, 5015, 6754, 7761, 8364 (if approved)
Intended status: Standards Track
Expires: December 29, 2018

S. Venaas
Cisco Systems, Inc.
A. Retana
Huawei R&D USA

June 27, 2018

PIM reserved bits and type space extension
draft-venaas-pim-reserved-bits-01

Abstract

The currently defined PIM version 2 messages share a common message header format. The common header definition contains eight reserved bits. This document specifies how these bits may be used by individual message types, and creates a registry containing the per message type usage. This document also extends the PIM type space by defining three new message types. For each of the new types, four of the previously reserved bits are used to form an extended type range.

This document Updates RFC7761 and RFC3973 by defining the use of the currently Reserved field in the PIM common header. This document further updates RFC7761 and RFC3973, along with RFC5015, RFC6754 and RFC8364, by specifying the use of the currently Reserved bits for each PIM message.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The currently defined PIM version 2 messages share a common message header format defined in the PIM Sparse Mode [RFC7761] and Dense Mode [RFC3973] specifications. The common header definition contains eight reserved bits. The message types defined in these documents all use this common header. However, several messages already make use of one or more bits, including the Bootstrap [RFC5059], DF-Election [RFC5015], and PIM Flooding Mechanism (PFM) [RFC8364] messages. There is no document formally specifying that these bits are to be used per message type.

This document refers to the bits specified as Reserved in the common PIM header [RFC7761] [RFC3973] as PIM message type flag bits, or simply flag bits, and it specifies that they are to be separately used on a per message type basis. It creates a registry containing the the per message type usage. For a particular message type, the usage of the flag bits can be defined in the document defining the message type, or a new document that updates that document.

The PIM message types as defined in the PIM Sparse Mode [RFC7761] and Dense Mode [RFC3973] specifications are in the range from 0 to 15. That type space is almost exhausted. Message type 15 was reserved by [RFC6166] for type space extension. In Section 5, this document specifies the use of the flag bits for message types 13, 14 and 15 in order to extend the PIM type space. The registration procedure for the extended type space is the same as for the existing type space, and the existing PIM message type registry is updated to include the extended type space.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. PIM header common format

The common PIM header is defined in section 4.9 of [RFC7761] and section 4.7.1 of [RFC3973]. This document updates the definition of the Reserved field and refers to that field as PIM message type flag bits, or simply flag bits. The new common header format is as below.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|PIM Ver| Type  |   Flags Bits  |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Flags Bits field is defined in Section 4. All other fields remain unchanged.

4. Flag Bit definitions

Unless otherwise specified, all the flag bits for each PIM type are Reserved [RFC8126]. They MUST be set to zero on transmission, and they MUST be ignored upon receipt. The specification of a new PIM type, MUST indicate whether the bits should be treated differently.

Currently for the message types 0 (Hello), 1 (Register), 2 (Register Stop), 3 (Join/Prune), 5 (Assert), 6 (Graft), 7 (Graft-Ack), 8 (Candidate RP Advertisement), 9 (State Refresh) and 11 (ECMP Redirect), all flag bits are Reserved.

When defining flag bits it is helpful to have a well defined way of referring to a particular bit. The most significant of the flag bits, the bit immediately following the type field is referred to as bit 7. The least significant, the bit right in front of the checksum field is referred to as bit 0. This is shown in the diagram below.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|PIM Ver| Type  |7 6 5 4 3 2 1 0|           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

4.1. Flag Bits for Type 4 (Bootstrap)

PIM message type 4 (Bootstrap) [RFC5059] defines flag bit 7 as No-Forward. The usage of the bit is defined in that document. The remaining flag bits are Reserved.

4.2. Flag Bits for Type 10 (DF Election)

PIM message type 10 (DF Election) [RFC5015] specifies that the four most significant flag bits (bits 4-7) are to be used as a sub-type. The remaining flag bits are currently Reserved.

4.3. Flag Bits for Type 12 (PFM)

PIM message type 12 (PFM) [RFC8364] defines flag bit 7 as No-Forward. The usage of the bit is defined in that document. The remaining flag bits are Reserved.

4.4. Flag Bits for Type 13 (Type Space Extension)

This type and the flag bit usage is defined in Section 5.

4.5. Flag Bits for Type 14 (Type Space Extension)

This type and the flag bit usage is defined in Section 5.

4.6. Flag Bits for Type 15 (Type Space Extension)

This type and the flag bit usage is defined in Section 5.

5. PIM Type Space Extension

The type space defined by the existing PIM specifications is almost exhausted. This document defines types 13, 14 and 15 (Type Space Extension) allowing for 48 additional types by for each of the three types, using the four most significant flag bits (bits 4-7) as a new field to store the extended type. These types are referred to as types 13.0 to 13.15, 14.0 to 14.15 and 15.0 to 15.15 where the last number denotes the value stored in the new field. The remaining four flag bits (bits 0-3) are Reserved to be used by each extended type. The specification of a new PIM extended type MUST indicate whether the bits should be treated differently. The common header for the new types is shown in the diagram below. The "Type" field is set to 13, 14 or 15, and the extended type field "SubType" denotes the value after the dot.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |PIM Ver| Type  |SubType| Rsvd  |           Checksum            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6. Security Considerations

This document clarifies the use of the flag bits in the common PIM header and it extends the PIM type space. As such, there is no impact on security or changes to the considerations in [RFC7761] and [RFC3973].

7. IANA considerations

This document updates the PIM Message Types registry and also creates a PIM Message Type Flag Bits registry that shows which flag bits are defined for use by each of the PIM message types.

The following changes should be made to the existing PIM Message Types registry. For types 4 (Bootstrap) and 8 (Candidate RP Advertisement) a reference to RFC5059 should be added. For the currently unassigned types 13 and 14, and the reserved type 15, the name should be changed to "Type Space Extension", and reference this document. In addition, right underneath each of the rows for types 13, 14 and 15, there should be a new row where it says "13.0-13.15 Unassigned", "14.0-14.15 Unassigned" and "15.0-15.15 Unassigned", respectively.

A new registry called "PIM Message Type Flag Bits" should be created in the pim-paremeters section with registration procedure "IETF

Review" as defined in [RFC8126] with this document as a reference. The initial content of the registry should be as below.

Type	bit(s)	Name	Reference
0	0-7	Reserved	[RFC3973][RFC7761]
1	0-7	Reserved	[RFC3973][RFC7761]
2	0-7	Reserved	[RFC3973][RFC7761]
3	0-7	Reserved	[RFC3973][RFC7761]
4	0-6	Reserved	[RFC3973][RFC7761]
4	7	No-Forward	[RFC5059]
5	0-7	Reserved	[RFC3973][RFC7761]
6	0-7	Reserved	[RFC3973][RFC7761]
7	0-7	Reserved	[RFC3973][RFC7761]
8	0-7	Reserved	[RFC3973][RFC7761]
9	0-7	Reserved	[RFC3973][RFC7761]
10	0-3	Reserved	[RFC3973][RFC7761]
10	4-7	Sub-type	[RFC5015]
11	0-7	Reserved	[RFC6754]
12	0-6	Reserved	[RFC3973][RFC7761]
12	7	No-Forward	[RFC8364]
13	0-3	N/A (used by 13.0-13.15)	[this document]
13	4-7	Extended type	[this document]
13.0-13.15	0-3	Reserved	[this document]
14	0-3	N/A (used by 14.0-14.15)	[this document]
14	4-7	Extended type	[this document]
14.0-14.15	0-3	Reserved	[this document]
15	0-3	N/A (used by 15.0-15.15)	[this document]
15	4-7	Extended type	[this document]
15.0-15.15	0-3	Reserved	[this document]

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3973] Adams, A., Nicholas, J., and W. Siadak, "Protocol Independent Multicast - Dense Mode (PIM-DM): Protocol Specification (Revised)", RFC 3973, DOI 10.17487/RFC3973, January 2005, <https://www.rfc-editor.org/info/rfc3973>.

[RFC5015] Handley, M., Kouvelas, I., Speakman, T., and L. Vicisano, "Bidirectional Protocol Independent Multicast (BIDIR-PIM)", RFC 5015, DOI 10.17487/RFC5015, October 2007, <https://www.rfc-editor.org/info/rfc5015>.

[RFC5059] Bhaskar, N., Gall, A., Lingard, J., and S. Venaas, "Bootstrap Router (BSR) Mechanism for Protocol Independent Multicast (PIM)", RFC 5059, DOI 10.17487/RFC5059, January 2008, <https://www.rfc-editor.org/info/rfc5059>.

[RFC6754] Cai, Y., Wei, L., Ou, H., Arya, V., and S. Jethwani, "Protocol Independent Multicast Equal-Cost Multipath (ECMP) Redirect", RFC 6754, DOI 10.17487/RFC6754, October 2012, <https://www.rfc-editor.org/info/rfc6754>.

[RFC7761] Fenner, B., Handley, M., Holbrook, H., Kouvelas, I., Parekh, R., Zhang, Z., and L. Zheng, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", STD 83, RFC 7761, DOI 10.17487/RFC7761, March 2016, <https://www.rfc-editor.org/info/rfc7761>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8364] Wijnands, IJ., Venaas, S., Brig, M., and A. Jonasson, "PIM Flooding Mechanism (PFM) and Source Discovery (SD)", RFC 8364, DOI 10.17487/RFC8364, March 2018, <https://www.rfc-editor.org/info/rfc8364>.

8.2. Informative References

[RFC6166] Venaas, S., "A Registry for PIM Message Types", RFC 6166, DOI 10.17487/RFC6166, April 2011, <https://www.rfc-editor.org/info/rfc6166>.

Authors’ Addresses

Stig Venaas
Cisco Systems, Inc.
Tasman Drive
San Jose CA 95134
USA

Email: stig@cisco.com

Alvaro Retana
Huawei R&D USA
2330 Central Expressway
Santa Clara CA 95050
USA

Email: alvaro.retana@huawei.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 6, 2018

J. Xie
Y. Liu
M. McBride
Huawei Technologies

March 5, 2018

PIM Extensions for P2MP-based BIER
draft-xie-pim-bier-extension-00

Abstract

Bit Index Explicit Replication (BIER) is a new architecture that provides optimal multicast forwarding without requiring intermediate routers to maintain any per-flow state by using a multicast-specific BIER header. PIM is a well-known multicast-specific routing protocol which is widely deployed either in a VRF context or in a Non-VRF context. This document describes PIM extensions to signal a P2MP Tree with BIER information, which is called a P2MP based BIER in [I.D.xie-bier-mvpn-mpls-p2mp]. PIM is required to alloc Label to build a P2MP tree hop-by-hop, and build a P2MP based BIER forwarding table further. This requires a BitMask being carried as a PIM Join Attribute by downstream node to upstream node hop-by-hop, and the behavior is like precedures as specified in [RFC6807].

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

Status of This Memo

Table of Contents

1. Introduction

Bit Index Explicit Replication (BIER) is a new architecture that provides optimal multicast forwarding without requiring intermediate routers to maintain any per-flow state by using a multicast-specific BIER header. PIM defined in [RFC7761] is a well-known multicast-specific routing protocol which is widely deployed either in a VRF context or in a Non-VRF context. This document describes PIM extensions to signal a P2MP Tree with BIER information, which is called a P2MP based BIER in [I-D.xie-bier-mvpn-mpls-p2mp], in which PIM is required to alloc Label to build a P2MP tree hop-by-hop, and build a P2MP based BIER forwarding table further. This requires a BitMask being carried as a PIM Join Attribute by downstream node to upstream node hop-by-hop, and the behavior is like precedures as specified in [RFC6807].

This document defines support for MPLS encapsulation as specified in [RFC8296]. Support for other encapsulation types is outside the scope of this document. The use of multiple encapsulation types is outside the scope of this document.

2. Terminology

Readers of this document are assumed to be familiar with the terminology and concepts of the documents listed as Normative References. For convenience, some of the more frequently used terms and new terms list below.

- o BFR: BIER Forwarding Router
- o BFR-ID: BIER Forwarding Router IDentify.
- o P2MP: Point to Multi-point
- o P2MP based BIER: BIER using P2MP as topology
- o F-BM: Forwarding Bit Mask
- o BSL: Bit String Length, that is 64, 128, 256, etc (per [RFC8279]).

3. PIM Signaling a P2MP BIER tree

3.1. Example of signaling the P2MP-BIER

Consider LSRs A - F, interconnected as follows:

```
( A ) ------------ ( B ) ------------ ( C ) ------------ ( D )
(Root)               \                  \           1 (0:0001)
                      \                  \
                     ( E )              ( F )
                   3 (0:0100)         2 (0:0010)
```

Figure 1: P2MP-based BIER Topology

Say that the node D has a BFR-id 1, F has a BFR-id 2, and E has a BFR-id 3, and we use a Bit String Length 4 (which is not valid per [RFC8296]) as an example.

Consider a target PIM tree identified by <S=RootAddress, G>, for which A is the Root, and say that D,E,F are all the Leafs of this PIM tree.

When D join the PIM tree, it alloc a Label, and bring this label with a F-BM of 0001 in a PIM Join attribute, and send it to the upstream node C.

When F join the PIM tree, it alloc a Label, and bring this label with a F-BM of 0010 in a PIM Join attribute, and send it to the upstream node C.

When E join the PIM tree, it alloc a Label, and bring this label with a F-BM of 0100 in a PIM Join attribute, and send it to the upstream node B.

When C get the PIM join messages from D and F, then C will establish a PIM state (S,G) and one or many downstream states, C also establish a PIM upstream state and send PIM Join to its upstream neighbor B, with a new allocated Label and a F-BM of 0011.

When B get the PIM join messages from E and C, then B will establish a PIM state (S,G) and one or many downstreams, B also establish a PIM upstream state and send PIM Join to it’s upstream neighbor A, with a new allocated Label and a F-BM of 0111.

When A get the PIM join message from B, A will establish a PIM state (S,G) and the downstream(s).

Each node of the PIM tree will establish a routing state of PIM (S,G), and a forwarding state of P2MP based BIER. Here we list the forwarding state of P2MP based BIER on every node.

A:

NHLFE (TreeID, OutInterface<to B>, OutLabel<alloc by B>, F-BM=0111)

B:

ILM(inLabel<alloc by B>, action<Replication to TreeID>, flag=CheckBS|Branch, BSL)

NHLFE (TreeID, OutInterface<to C>, OutLabel<alloc by C>, F-BM=0011)

NHLFE (TreeID, OutInterface<to E>, OutLabel<alloc by E>, F-BM=0100)

C:

ILM(inLabel<alloc by C>, action<Replication to TreeID>,
flag=CheckBS|Branch, BSL)

NHLFE (TreeID, OutInterface<to D>, OutLabel<alloc by D>,
F-BM=0001)

NHLFE (TreeID, OutInterface<to F>, OutLabel<alloc by F>,
F-BM=0100)

E:

ILM(inLabel<alloc by E>, action<Replication to TreeID>,
flag=CheckBS|Leaf, BSL)

LEAF(TreeID, F-BM=0100, Flag=PopBIERincluding)

D:

ILM(inLabel<alloc by D>, action<Replication to TreeID>,
flag=CheckBS|Leaf, BSL)

LEAF(TreeID, F-BM=0001, Flag=PopBIERincluding)

F:

ILM(inLabel<alloc by F>, action<Replication to TreeID>,
flag=CheckBS|Leaf, BSL)

LEAF(TreeID, F-BM=0010, Flag=PopBIERincluding)

3.2. BIER-Supported Hello Option

A PIM router indicates that it supports the mechanism specified in this document by including the BIER-Signal-Supported Hello option in its PIM Hello message. Note that it also needs to include the Join Attribute Hello option as specified in [RFC5384]. The format of the BIER-Signal-Supported Hello option is defined to be:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          OptionType           |         OptionLength          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|P|D|I|R|       Reserved        |         Reserved              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 P: The node has BIER P-Capability.
 D: The node has BIER D-Capability.
 I: The node Ignore the BIER Header except the Label.
 R: The node Require a packet without BIER Header except the Label.
```

Figure 2: BIER-Supported Hello Option

OptionType = TBD, OptionLength = 4.

P-Capability indicate a complete BIER function, which includes P-Capability and D-Capability. If a node support P-Capability, then it support the whole BIER function, which means it support both P-capability and D-capability.

D-Capability indicate a subset of BIER function, to Disposit BIER Header of a packet including or excluding the BIER Label. If a node doesn’t support P-Capability, it may still support D-Capability. If a node don’t support D-capability, it is supposed not to support P-Capability.

If a Node doesn’t have P-Capability, then P flag MUST be cleared. Whether the node will be a Branch or BUD or Leaf, the I flag SHOULD be set.

If a node doesn’t have D-Capability, then P and D flag MUST be cleared. If the node will be a BUD or Leaf then R flag SHOULD be set. if the node will be a Branch then R flag MAY not be set.

If a node doesn’t have P-Capability but does have D-Capability, then D flag SHOULD be set, but R flag MAY be set or not be set.

3.3. New BIER F-BM Join Attribute Format

When a PIM router supports this mechanism and has determined from a received Hello that the neighbor supports this mechanism, and also that all the neighbors on the interface support the use of join attributes, it will send Join/Prune messages that MAY include a BIER F-BM Join Attribute. The mechanism to process a PIM Join Attribute is described in [RFC5384]. The format of the new attribute is specified in the following.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|F|E| Attr_Type |    Length     |Reserve|BS Len |Set Identifier |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  F-BM  (first 32 bits)                        ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                                                               ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                  F-BM  (last 32 bits)                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: BIER F-BM Join Attribute

F bit: 0 (Non-Transitive Attribute).

E bit: As specified by [RFC5384].

Attr_Type: TBD.

Length: The minimum length is 10, that is a 2 octets BSL and a minimum of 8 octets (64 bits).

BS Len: Bit String Length, that is the Length of F-BM

Set Identifier: As defined in [RFC8279].

F-BM: As defined in [RFC8279].

4. How to Use BIER F-BM Join Attribute

A router supporting this mechanism MUST, unless administratively disabled, include the PIM Join Attribute option in its PIM Hellos. See [RFC5384] and "PIM-Hello Options" on [PIM-REG] for details.

It is RECOMMENDED that implementations allow for administrative control of whether to make use of this mechanism. Implementations MAY also allow further control of what information to store and send upstream, by configuring whether the node require a packet without BIER header.

It is important to note that when a node’s downstream F-BM OR’ing result changed, it SHOULD trigger a new Join message with an updated BIER F-BM Join Attribute.

When a router removes a link from an oif-list, it needs to be able to reevaluate the BIER F-BM that it will advertise upstream. This happens when an oif-list entry is timed out or a Prune is received.

It is RECOMMENDED that the Join Attribute defined in this document be used only for entries in the join-list part of the Join/Prune message. If the attribute is used in the prune-list, an implementation MUST ignore it and process the Prune as if the attribute were not present.

It is also RECOMMENDED that join suppression be disabled on a LAN when BIER F-BM is used.

It is RECOMMENDED that, when triggered Join/Prune messages are sent by a downstream router, the BIER F-BM information not be included in the message. This way, when convergence is important, avoiding the processing time to build an BIER F-BM record in a downstream router and processing time to parse the message in the upstream router will help reduce convergence time. If an upstream router receives a Join/Prune message with no BIER F-BM data, it SHOULD NOT interpret the message as a trigger to clear or reset the BIER F-BM data it has cached.

5. Capability and Error Handling

TBD.

6. IANA Considerations

Allocation is expected from IANA for codepoints from the "PIM-Hello Options" registry and the "PIM Join Attribute Types" registry.

7. Security Considerations

TBD

8. Acknowledgements

TBD

9. References

9.1. Normative References

[I-D.ietf-bier-mvpn]
Rosen, E., Sivakumar, M., Aldrin, S., Dolganow, A., and T. Przygienda, "Multicast VPN Using BIER", draft-ietf-bier-mvpn-10 (work in progress), February 2018.

[I-D.xie-bier-mvpn-mpls-p2mp]
Xie, J., "Multicast VPN Using MPLS P2MP and BIER", draft-xie-bier-mvpn-mpls-p2mp-00 (work in progress), October 2017.

[RFC5384] Boers, A., Wijnands, I., and E. Rosen, "The Protocol Independent Multicast (PIM) Join Attribute Format", RFC 5384, DOI 10.17487/RFC5384, November 2008, <https://www.rfc-editor.org/info/rfc5384>.

[RFC6807] Farinacci, D., Shepherd, G., Venaas, S., and Y. Cai, "Population Count Extensions to Protocol Independent Multicast (PIM)", RFC 6807, DOI 10.17487/RFC6807, December 2012, <https://www.rfc-editor.org/info/rfc6807>.

[RFC7761] Fenner, B., Handley, M., Holbrook, H., Kouvelas, I., Parekh, R., Zhang, Z., and L. Zheng, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", STD 83, RFC 7761, DOI 10.17487/RFC7761, March 2016, <https://www.rfc-editor.org/info/rfc7761>.

[RFC8279] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Przygienda, T., and S. Aldrin, "Multicast Using Bit Index Explicit Replication (BIER)", RFC 8279, DOI 10.17487/RFC8279, November 2017, <https://www.rfc-editor.org/info/rfc8279>.

[RFC8296] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Tantsura, J., Aldrin, S., and I. Meilik, "Encapsulation for Bit Index Explicit Replication (BIER) in MPLS and Non-MPLS Networks", RFC 8296, DOI 10.17487/RFC8296, January 2018, <https://www.rfc-editor.org/info/rfc8296>.

9.2. Informative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

Authors’ Addresses

Jingrong Xie
Huawei Technologies
Q15 Huawei Campus, No.156 Beiqing Rd.
Beijing 100095
China

Email: xiejingrong@huawei.com

Yisong Liu
Huawei Technologies

Email: liuyisong@huawei.com

Mike McBride
Huawei Technologies

Email: mmcbride7@gmail.com