PIM Working Group
Internet Draft
Intended status: Standards Track
Expires: August 7, 2018

H. Zhao
Ericsson
X. Liu
Jabil
Y. Liu
Huawei
M. Sivakumar
Cisco
A. Peter
Individual

February 8, 2018

A Yang Data Model for IGMP and MLD Snooping
draft-ietf-pim-igmp-mld-snooping-yang-00

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.

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:

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

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

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

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

o 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 id?                                 uint32

    |     +--rw type?                               enumeration

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

    |     +--rw forwarding-mode?                    enumeration

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

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

```
       |     +--rw send-query?                          boolean
       |     +--rw fast-leave?                          empty {fast-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-
l2-
   multicast-group}?
       |     +--rw static-vpls-mrouter-interface*       l2vpn-instance-pw-ref
{static-
   l2-multicast-group}?
       |     +--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 vpls-outgoing-ac*              l2vpn-instance-ac-ref
       |     |  +--rw vpls-outgoing-pw*              l2vpn-instance-pw-ref
       |     +--ro entries-count?                       uint32
       |     +--ro bridge-mrouter-interface*            if:interface-ref
       |     +--ro vpls-mrouter-interface*              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 vpls-outgoing-ac*            l2vpn-instance-ac-ref
       |     |     +--ro vpls-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
       |     +--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 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 pim?                    yang:counter64
       +--rw mld-snooping-instances
       |  +--rw mld-snooping-instance* [name]
       |     +--rw name                                string
       |     +--rw id?                                 uint32
       |     +--rw type?                               enumeration
       |     +--rw enable?                             boolean {admin-enable}?
       |     +--rw forwarding-mode?                    enumeration
       |     +--rw explicit-tracking?                  boolean {explicit-
tracking}?
       |     +--rw exclude-lite?                       boolean {exclude-lite}?
       |     +--rw send-query?                         boolean
       |     +--rw fast-leave?                         empty {fast-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-
l2-
   multicast-group}?

       |     +--rw static-vpls-mrouter-interface*     l2vpn-instance-pw-ref
{static-
   l2-multicast-group}?

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

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

       |     +--ro entries-count?                       uint32

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

       |     +--ro vpls-mrouter-interface*              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 vpls-outgoing-ac*            l2vpn-instance-ac-ref
```

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

2.3. IGMP and MLD Snooping References

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

The type of the instance indicates the scenario which is bridge or VPLS. When referenced in bridge, the id of instance means VLAN id. When referenced in VPLS, the id means VSI id.

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

...

+--rw bridges

    |  +--rw bridge* [name]

    |     +--rw name                      dot1qtypes: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                       dot1qtypes:vlan-index-type

    |              +--rw igmp-snooping-instance?   igmp-snooping-instance-ref

    |              +--rw mld-snooping-instance?    mld-snooping-instance-ref

    |              +--rw interfaces

    |                 +--rw interface* [name]

    |                    +--rw name                      string

    |                    +--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

          +--rw endpoint* [name]

             +--rw name                      string

             +--rw igmp-snooping-instance?   igmp-snooping-instance-ref

             +--rw mld-snooping-instance?    mld-snooping-instance-ref

             +--rw (ac-or-pw-or-redundancy-grp)?

                +--:(ac)

                |  +--rw ac* [name]

                |     +--rw name                      string

                |     +--rw igmp-snooping-instance?   igmp-snooping-instance-ref

                |     +--rw mld-snooping-instance?    mld-snooping-instance-ref

                +--:(pw)

                |  +--rw pw* [name]

                |     +--rw name                      string

                |     +--rw igmp-snooping-instance?   igmp-snooping-instance-ref

                |     +--rw mld-snooping-instance?    mld-snooping-instance-ref

                +--:(redundancy-grp)

                   +--rw (primary)

                   |  +--:(primary-ac)
```

```
                  |  |  +--rw primary-ac
                  |  |     +--rw name?                     string
                  |  |     +--rw igmp-snooping-instance?   igmp-snooping-
instance-ref
                  |  |     +--rw mld-snooping-instance?    mld-snooping-
instance-ref
                  |  +--:(primary-pw)
                  |     +--rw primary-pw* [name]
                  |        +--rw name                     string
                  |        +--rw igmp-snooping-instance?   igmp-snooping-
instance-ref
                  |        +--rw mld-snooping-instance?    mld-snooping-
instance-ref
                  +--rw (backup)?
                     +--:(backup-ac)
                     |  +--rw backup-ac
                     |     +--rw name?                     string
                     |     +--rw igmp-snooping-instance?   igmp-snooping-
instance-ref
                     |     +--rw mld-snooping-instance?    mld-snooping-
instance-ref
                     +--:(backup-pw)
                        +--rw backup-pw* [name]
                           +--rw name                     string
                           +--rw igmp-snooping-instance?   igmp-snooping-
instance-ref
                           +--rw mld-snooping-instance?    mld-snooping-
instance-ref
```

2.4. 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 id?       uint32
       |     +---w group?    inet:ipv4-address
       |     +---w source?   inet:ipv4-address
       +---x clear-mld-snooping-groups {rpc-clear-groups}?
          +---w input
             +---w id?       uint32
             +---w group?    inet:ipv6-address
             +---w source?   inet:ipv6-address
```

3. IGMP and MLD Snooping YANG Module

```
<CODE BEGINS> file "ietf-igmp-mld-snooping@2017-10-25.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";
  }


   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:mmcbride7@gmail.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 2017-10-25 {
         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 fast-leave {
        description
          "Support configuration of fast-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 per-instance-config {
        description
          "Support configuration of each VLAN or VPLS 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 "/l2vpn:l2vpn/l2vpn:instances" +
            "/l2vpn:instance/l2vpn:endpoint/l2vpn:ac/l2vpn:name";
      }
      description "l2vpn-instance-ac-ref";
    }

    typedef l2vpn-instance-pw-ref {
      type leafref {
         path "/l2vpn:l2vpn/l2vpn:instances" +
            "/l2vpn:instance/l2vpn:endpoint/l2vpn:pw/l2vpn:name";
      }
      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  "Statistics of IGMP and MLD Snooping ";

        container statistics {
          config false;
          description
             "The statistics of IGMP and MLD Snooping related packets.";

          container received {
            description "Statistics of received messages.";
            uses general-statistics-sent-received;
          }
          container sent {
            description "Statistics of sent messages.";
            uses general-statistics-sent-received;
          }
        } // statistics

      } // general-state-attributes
```

```
      grouping instance-config-attributes-igmp-snooping {
        description "IGMP snooping configuration for each VLAN or VPLS
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 "ims:type = 'bridge'";
            type if:interface-ref;
            description "Outgoing interface in bridge fowarding";

         }

         leaf-list vpls-outgoing-ac {
            when "ims:type = 'vpls'";
            type l2vpn-instance-ac-ref;
            description "Outgoing ac in vpls fowarding";

         }
```

```
         leaf-list vpls-outgoing-pw {
            when "ims:type = 'vpls'";
            type l2vpn-instance-pw-ref;
            description "Outgoing pw in vpls 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 fast-leave {
          if-feature fast-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 "ims:type = 'bridge'";
         if-feature static-l2-multicast-group;
         type if:interface-ref;
         description "static mrouter interface in bridge fowarding";

      }


      leaf-list static-vpls-mrouter-interface {

         when "ims:type = 'vpls'";
         if-feature static-l2-multicast-group;
         type l2vpn-instance-pw-ref;
         description "static mrouter interface in vpls fowarding";

      }


      } // 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 "ims:type = 'bridge'";
            type if:interface-ref;
            description "Outgoing interface in bridge fowarding";

         }

         leaf-list vpls-outgoing-ac {
            when "ims:type = 'vpls'";
            type l2vpn-instance-ac-ref;
            description "Outgoing ac in vpls fowarding";

         }

         leaf-list vpls-outgoing-pw {
            when "ims:type = 'vpls'";
            type l2vpn-instance-pw-ref;
            description "Outgoing pw in vpls 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 VPLS
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

      // statistics
      uses general-state-attributes;


      } // 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 VPLS 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 "ims:type = 'bridge'";
         type if:interface-ref;
         config false;
         description " mrouter interface in bridge fowarding";

        }



        leaf-list vpls-mrouter-interface {

         when "ims:type = 'vpls'";
         type l2vpn-instance-pw-ref;
         config false;
         description " mrouter interface in vpls 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

      // statistics
      uses general-state-attributes;

      } // 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 VPLS instance or EVPN instance.";
```

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

      }

      leaf-list vpls-outgoing-ac {
         when "ims:type = 'vpls'";
         type l2vpn-instance-ac-ref;
         description "Outgoing ac in vpls fowarding";

      }

      leaf-list vpls-outgoing-pw {
         when "ims:type = 'vpls'";
         type l2vpn-instance-pw-ref;
         description "Outgoing pw in vpls 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 pim {
          type yang:counter64;
          description
            "The number of pim hello messages.";
```

```
        }
      } // general-statistics-sent-received


   grouping endpoint-grp {
      description "A grouping that defines the structure of " +
                "an endpoint";
      choice ac-or-pw-or-redundancy-grp {
         description "A choice of attachment circuit or " +
                   "pseudowire or redundancy group";
         case ac {
            description "Attachment circuit(s) as an endpoint";
            list ac {
               key "name";
               leaf name {
                  type string;
                  description "Name of attachment circuit.  " +
                            "This field is intended to " +
                            "reference standardized " +
                            "layer-2 definitions.";
               }
               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";
               }

               description "An L2VPN instance's " +
                         "attachment circuit list";
            }
         }
         case pw {
            description "Pseudowire(s) as an endpoint";
            list pw {
               key "name";
               leaf name {
                  type string;
                  description "Name of Pseudowire.";
               }
               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";
               }

               description "An L2VPN instance's " +
                        "pseudowire(s) list";
            }
         }
         case redundancy-grp {
            description "Redundancy group as an endpoint";
            choice primary {
               mandatory true;
               description "primary options";
               case primary-ac {
                  description "primary-ac";
                  container primary-ac {
                     description "Primary AC";
                     leaf name {
                        type string;
                        description "Name of attachment circuit.  ";
                     }
                     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";
                     }
                  }// primary-ac
               }// primary-ac

               case primary-pw {

                  list primary-pw {
                     key "name";
                     leaf name {
                        type string;
                        description "Name of Pseudowire.";
                     }
```

```
                     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";
                     }

                  description "primary-pw";

                  } //primary-pw
               }//primary-pw
            }
            choice backup {
               description "backup options";
               case backup-ac {
                  description "backup-ac";
                  container backup-ac {
                     description "Backup AC";
                     leaf name {
                        type string;
                        description "Name of attachment circuit.  ";
                     }
                     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";
                     }
                  }// backup-ac
               }// backup-ac
               case backup-pw {
                  description "backup-pw";
                  list backup-pw {
                     key "name";
                     leaf name {
                        type string;
                        description "Name of Pseudowire.";
                     }
                     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";
                     }

                     description "backup-pw";
                  } //backup-pw
               }
            }
         }
      }
   }


      /*
       * 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 id {
        type uint32;
        description
         "It is vlan_id or vpls_id.
         When igmp-snooping-instance is applied under bridge view, its
value is 0.";
          }
```

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


       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 id {
```

```
        type uint32;
        description
         "It is vlan_id or vpls_id.
         When mld-snooping-instance is applied under bridge view, its
value is 0.";
         }

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


       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";
                     }
                     container interfaces {
                        description
                        "Interface configuration parameters.";
```

```
                        list interface {
                           key "name";

                           description
                              "The list of configured interfaces on the
device.";

                           leaf name {
                              type string;
                              description
                              "The name of the interface.";
                           }
                           leaf igmp-snooping-instance {
                              type igmp-snooping-instance-ref;
                              description "Configure igmp-snooping
instance under the interface view";
                           }
                           leaf mld-snooping-instance {
                              type mld-snooping-instance-ref;
                              description "Configure mld-snooping
instance under the interface view";
                           }
                        }
                     }//interfaces
                  }//vlan
               }//bridge-vlan
            }//component
         }//bridge
     }   //bridges




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

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

         leaf name {
               type string;
               description "Name of VPLS 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";
         }

         list endpoint {
            key "name";
            description "An endpoint";
            leaf name {
               type string;
               description "endpoint name";
            }
            leaf igmp-snooping-instance {
               type igmp-snooping-instance-ref;
               description "Configure igmp-snooping instance under the
interface view";
            }
            leaf mld-snooping-instance {
               type mld-snooping-instance-ref;
               description "Configure mld-snooping instance under the
interface view";
            }

            uses endpoint-grp;

         } //endpoint
      }
      }

      /*
       * RPCs
       */

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

        input {
          leaf id {
                type uint32;
```

```
                description
                  "VLAN ID, VPLS ID, or EVPN ID";
          }

          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 id {
                type uint32;
                description
                  "VLAN ID, VPLS ID, or EVPN ID";
          }

          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