PIM Working Group
Internet Draft
Intended status: Standards Track
Expires: April 07, 2022

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

October 08, 2021

A Yang Data Model for IGMP and MLD Snooping
draft-ietf-pim-igmp-mld-snooping-yang-20.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).

Status of this Memo

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].

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]
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]. It also includes actions for clearing IGMP and MLD Snooping group tables.

The YANG module doesn’t cover L2VPN, which will be specified in a separated document.

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, and it corresponds to one BRIDGE instance.

Currently the value of l2-service-type in igmp-snooping-instance could only be set bridge. After it is set, igmp-snooping-instance could be used in the BRIDGE service.

The values of bridge-mrouter-interface is filled by the snooping device dynamically. It is different from static-bridge-mrouter-interface which is 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 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 static-bridge-mrouter-interface*   if:interface-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
       +--ro entries-count?                     yang:gauge32
       +--ro bridge-mrouter-interface*          if:interface-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 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* [address] {explicit-tracking}?
        |        +--ro address        inet:ipv4-address
        |        +--ro 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] 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, and it corresponds to one BRIDGE instance.

Currently the value of l2-service-type in mld-snooping-instance could only be set bridge. After it is set, mld-snooping-instance could be used in the BRIDGE service.

The value of bridge-mrouter-interface is filled by the snooping device dynamically. It is different from static-bridge-mrouter-interface which is 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 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 static-bridge-mrouter-interface*  if:interface-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
       +--ro entries-count?                    yang:gauge32
       +--ro bridge-mrouter-interface*         if:interface-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 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* [address] {explicit-tracking}?
       |        +--ro address        inet:ipv6-address
       |        +--ro 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] 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.

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

```
     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
```

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],[dot1Qcp].

```
<CODE BEGINS> file ietf-igmp-mld-snooping@2021-10-08.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 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) 2021 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 2021-10-08 {
    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 fast-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 static L2 multicast 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 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
        "When it is true, this switch will send out periodic
         IGMP General Query Message or MLD General Query Message.";
    }

    leaf fast-leave {
      if-feature fast-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. If it doesn't exist,
         the IGMP or MLD packet will be ignored.";
      reference "RFC 3376. Sec. 9.1, 9.2, 9.3.";
    }

    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";
    }
  } // 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 dynamically 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.";
    }
  } // 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 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
          "It indicates BRIDGE or other services.";
      }

      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
          "The source address of IGMP General Query message,
           which is sent out by this switch.";
      }

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

            leaf address {
              type inet:ipv4-address;
              description
                "Multicast membership host address.";
            }
            leaf 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
          "It indicates BRIDGE or other services.";
      }

      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
          "The source address of MLD General Query message,
           which is sent out by this switch.";
      }

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

            leaf address {
              type inet:ipv6-address;
              description
                "Multicast membership host address.";
            }
            leaf 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";
    }
  }
}
<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.

7.2. Informative References

[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

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": "*"
  }
}
```

Authors' Addresses

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