ï»¿

Internet Engineering Task Force (IETF)
Request for Comments: 8652
Category: Standards Track
ISSN: 2070-1721

X. Liu
Volta Networks
F. Guo
Huawei
M. Sivakumar
Juniper
P. McAllister
Metaswitch Networks
A. Peter
IP Infusion India

November 2019

A YANG Data Model for the Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD)

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) devices.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc8652.

Table of Contents

1. Introduction

YANG [RFC6020] [RFC7950] is a data definition language that was introduced to model the configuration and running state of a device managed using network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. YANG is now also being used as a component of wider management interfaces, such as command-line interfaces (CLIs).

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) devices. The protocol versions include IGMPv1 [RFC1112], IGMPv2 [RFC2236], IGMPv3 [RFC3376], MLDv1 [RFC2710], and MLDv2 [RFC3810]. The core features of the IGMP and MLD protocols are defined as required. Non-core features are defined as optional in the provided data model.

The YANG data model in this document conforms to the Network Management Datastore Architecture (NMDA) [RFC8342].

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 abbreviations are used in this document and the defined model:

IGMP: Internet Group Management Protocol [RFC3376].

MLD: Multicast Listener Discovery [RFC3810].

SSM: Source-Specific Multicast service model [RFC3569] [RFC4607].

1.2. Conventions Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.3. Tree Diagrams

Tree diagrams used in this document follow the notation defined in [RFC8340].

1.4. 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
yang	ietf-yang-types	[RFC6991]
inet	ietf-inet-types	[RFC6991]
if	ietf-interfaces	[RFC8343]
ip	ietf-ip	[RFC8344]
rt	ietf-routing	[RFC8349]
rt-types	ietf-routing-types	[RFC8294]
acl	ietf-access-control-list	[RFC8519]

Table 1: Prefixes and Corresponding YANG Modules

2. Design of the Data Model

2.1. Scope of Model

The model covers IGMPv1 [RFC1112], IGMPv2 [RFC2236], IGMPv3 [RFC3376], MLDv1 [RFC2710], and MLDv2 [RFC3810].

This model does not cover other IGMP- and MLD-related protocols such as IGMP/MLD Proxy [RFC4605] or IGMP/MLD Snooping [RFC4541] etc., which will be specified in separate documents.

This model can be used to configure and manage various versions of IGMP and MLD protocols. The operational state data and statistics can be retrieved by this model. Even though no protocol-specific notifications are defined in this model, the subscription and push mechanism defined in [RFC8639] and [RFC8641] can be implemented by the user to subscribe to notifications on the data nodes in this model.

The model contains all the basic configuration parameters to operate the protocols listed above. Depending on the implementation choices, some systems may not allow some of the advanced parameters to be configurable. The occasionally implemented parameters are modeled as optional features in this model, while the rarely implemented parameters are not included in this model and left for augmentation. This model can be extended, and it has been structured in a way that such extensions can be conveniently made.

The protocol parameters covered in this model can been seen from the model structure described in Section 3.

The protocol parameters that were considered but are not covered in this model are described in the following sections.

2.1.1. Parameters Not Covered at the Global Level

The configuration parameters and operational states not covered on an IGMP instance or an MLD instance are:

* Explicit tracking

* Maximum transmit rate

* Last member query count

* Other querier present time

* Send router alert

* Startup query interval

* Startup query count

2.1.2. Parameters Not Covered at the Interface Level

The configuration parameters and operational states not covered on an IGMP interface or an MLD interface are:

* Disable router alert check

* Drop IGMP version 1, IGMP version 2, or MLD version 1

* Last member query count

* Maximum number of sources

* Other querier present time

* Passive mode

* Promiscuous mode

* Query before immediate leave

* Send router alert

2.2. Optional Capabilities

This model is designed to represent the capabilities of IGMP and MLD devices with various specifications, including the basic capability subsets of the IGMP and MLD protocols. 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 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

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

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

* The model can support IGMP (IPv4), MLD (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 RFC 8344, which uses separate branches for IPv4 and IPv6.

* The separate branches for IGMP and MLD can accommodate their differences better and cleaner. 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     <= Augmented by this Model
                    ...
         |     +--rw mld      <= Augmented by this Model
                    ...
```

The "igmp" container instantiates an IGMP protocol of version IGMPv1, IGMPv2, or IGMPv3. The "mld" container instantiates an MLD protocol of version MLDv1 or MLDv2.

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].

A configuration data node is marked as mandatory only when its value must be provided by the user. Where nodes are not essential to protocol operation, they are marked as optional. Some other nodes are essential but have a default specified, so that they are also optional and need not be configured explicitly.

3.1. IGMP Configuration and Operational State

The IGMP data is modeled as a schema subtree augmenting the "control-plane-protocol" data node under "/rt:routing/rt:control-plane-protocols" in the module ietf-routing, following the convention described in [RFC8349]. The augmentation to the module ietf-routing allows this model to support multiple instances of IGMP, but a restriction MAY be added depending on the implementation and the device. The identity "igmp" is derived from the "rt:control-plane-protocol" base identity and indicates that a control-plane-protocol instance is IGMP.

The IGMP subtree is a three-level hierarchy structure as listed below:

Global level: Including IGMP configuration and operational state attributes for the entire IGMP protocol instance in this router.

Interface-global level: Including configuration data nodes that are applicable to all the interfaces whose corresponding nodes are not defined or not configured at the interface level. For such a node at the interface level, the system uses the same value of the corresponding node at the interface-global level.

Interface level: Including IGMP configuration and operational state attributes specific to the given interface. For a configuration node at the interface level, there may exist a corresponding configuration node with the same name at the interface-global level. The value configured on a node at the interface level overrides the value configured on the corresponding node at the interface-global level.

```
augment /rt:routing/rt:control-plane-protocols
          /rt:control-plane-protocol:
  +--rw igmp {feature-igmp}?
     +--rw global
     |  +--rw enabled?         boolean {global-admin-enable}?
     |  +--rw max-entries?     uint32 {global-max-entries}?
     |  +--rw max-groups?      uint32 {global-max-groups}?
     |  +--ro entries-count?   uint32
     |  +--ro groups-count?    uint32
     |  +--ro statistics
     |     +--ro discontinuity-time?   yang:date-and-time
     |     +--ro error
     |     |  +--ro total?       yang:counter64
     |     |  +--ro query?       yang:counter64
     |     |  +--ro report?      yang:counter64
     |     |  +--ro leave?       yang:counter64
     |     |  +--ro checksum?    yang:counter64
     |     |  +--ro too-short?   yang:counter64
     |     +--ro received
     |     |  +--ro total?    yang:counter64
     |     |  +--ro query?    yang:counter64
     |     |  +--ro report?   yang:counter64
     |     |  +--ro leave?    yang:counter64
     |     +--ro sent
     |        +--ro total?    yang:counter64
     |        +--ro query?    yang:counter64
     |        +--ro report?   yang:counter64
     |        +--ro leave?    yang:counter64
     +--rw interfaces
        +--rw last-member-query-interval?   uint16
        +--rw query-interval?               uint16
        +--rw query-max-response-time?      uint16
        +--rw require-router-alert?         boolean
        |       {intf-require-router-alert}?
        +--rw robustness-variable?          uint8
        +--rw version?                      uint8
        +--rw max-groups-per-interface?     uint32
        |       {intf-max-groups}?
        +--rw interface* [interface-name]
           +--rw interface-name                 if:interface-ref
           +--rw last-member-query-interval?    uint16
           +--rw query-interval?                uint16
           +--rw query-max-response-time?       uint16
           +--rw require-router-alert?          boolean
           |       {intf-require-router-alert}?
           +--rw robustness-variable?           uint8
           +--rw version?                       uint8
           +--rw enabled?                       boolean
           |       {intf-admin-enable}?
           +--rw group-policy?
           |       -> /acl:acls/acl/name
           +--rw immediate-leave?               empty
           |       {intf-immediate-leave}?
           +--rw max-groups?                    uint32
           |       {intf-max-groups}?
           +--rw max-group-sources?             uint32
           |       {intf-max-group-sources}?
           +--rw source-policy?
           |       -> /acl:acls/acl/name {intf-source-policy}?
           +--rw verify-source-subnet?          empty
           |       {intf-verify-source-subnet}?
           +--rw explicit-tracking?             empty
           |       {intf-explicit-tracking}?
           +--rw lite-exclude-filter?           empty
           |       {intf-lite-exclude-filter}?
           +--rw join-group*
           |       rt-types:ipv4-multicast-group-address
           |       {intf-join-group}?
           +--rw ssm-map*
           |       [ssm-map-source-addr ssm-map-group-policy]
           |       {intf-ssm-map}?
```

```
|  +--rw ssm-map-source-addr     ssm-map-ipv4-addr-type
|  +--rw ssm-map-group-policy    string
+--rw static-group* [group-addr source-addr]
|       {intf-static-group}?
|  +--rw group-addr
|  |       rt-types:ipv4-multicast-group-address
|  +--rw source-addr
|          rt-types:ipv4-multicast-source-address
+--ro oper-status                    enumeration
+--ro querier                        inet:ipv4-address
+--ro joined-group*
|       rt-types:ipv4-multicast-group-address
|       {intf-join-group}?
+--ro group* [group-address]
   +--ro group-address
   |       rt-types:ipv4-multicast-group-address
   +--ro expire           uint32
   +--ro filter-mode      enumeration
   +--ro up-time          uint32
   +--ro last-reporter?   inet:ipv4-address
   +--ro source* [source-address]
      +--ro source-address    inet:ipv4-address
      +--ro expire            uint32
      +--ro up-time           uint32
      +--ro host-count?       uint32
      |       {intf-explicit-tracking}?
      +--ro last-reporter?    inet:ipv4-address
      +--ro host* [host-address]
              {intf-explicit-tracking}?
         +--ro host-address        inet:ipv4-address
         +--ro host-filter-mode    enumeration
```

3.2. MLD Configuration and Operational State

The MLD data is modeled as a schema subtree augmenting the "control-plane-protocol" data node under "/rt:routing/rt:control-plane-protocols" in the module ietf-routing, following the convention described in [RFC8349]. The augmentation to the module ietf-routing allows this model to support multiple instances of MLD, but a restriction MAY be added depending on the implementation and the device. The identity "mld" is derived from the "rt:control-plane-protocol" base identity and indicates that a control-plane-protocol instance is MLD.

The MLD subtree is a three-level hierarchy structure as listed below:

Global level: Including MLD configuration and operational state attributes for the entire MLD protocol instance in this router.

Interface-global level: Including configuration data nodes that are applicable to all the interfaces whose corresponding nodes are not defined or not configured at the interface level. For such a node at the interface level, the system uses the same value of the corresponding node at the interface-global level.

Interface level: Including MLD configuration and operational state attributes specific to the given interface. For a configuration node at the interface level, there may exist a corresponding configuration node with the same name at the interface-global level. The value configured on a node at the interface level overrides the value configured on the corresponding node at the interface-global level.

```
  augment /rt:routing/rt:control-plane-protocols
            /rt:control-plane-protocol:
    +--rw mld {feature-mld}?
       +--rw global
       |  +--rw enabled?         boolean {global-admin-enable}?
       |  +--rw max-entries?     uint32 {global-max-entries}?
       |  +--rw max-groups?      uint32 {global-max-groups}?
       |  +--ro entries-count?   uint32
```

```
|  +--ro groups-count?    uint32
|  +--ro statistics
|     +--ro discontinuity-time?   yang:date-and-time
|     +--ro error
|     |  +--ro total?       yang:counter64
|     |  +--ro query?       yang:counter64
|     |  +--ro report?      yang:counter64
|     |  +--ro leave?       yang:counter64
|     |  +--ro checksum?    yang:counter64
|     |  +--ro too-short?   yang:counter64
|     +--ro received
|     |  +--ro total?    yang:counter64
|     |  +--ro query?    yang:counter64
|     |  +--ro report?   yang:counter64
|     |  +--ro leave?    yang:counter64
|     +--ro sent
|        +--ro total?    yang:counter64
|        +--ro query?    yang:counter64
|        +--ro report?   yang:counter64
|        +--ro leave?    yang:counter64
|
+--rw interfaces
   +--rw last-member-query-interval?   uint16
   +--rw query-interval?               uint16
   +--rw query-max-response-time?      uint16
   +--rw require-router-alert?         boolean
   |       {intf-require-router-alert}?
   +--rw robustness-variable?          uint8
   +--rw version?                      uint8
   +--rw max-groups-per-interface?     uint32
   |       {intf-max-groups}?
   +--rw interface* [interface-name]
      +--rw interface-name                 if:interface-ref
      +--rw last-member-query-interval?    uint16
      +--rw query-interval?                uint16
      +--rw query-max-response-time?       uint16
      +--rw require-router-alert?          boolean
      |       {intf-require-router-alert}?
      +--rw robustness-variable?           uint8
      +--rw version?                       uint8
      +--rw enabled?                       boolean
      |       {intf-admin-enable}?
      +--rw group-policy?
      |       -> /acl:acls/acl/name
      +--rw immediate-leave?               empty
      |       {intf-immediate-leave}?
      +--rw max-groups?                    uint32
      |       {intf-max-groups}?
      +--rw max-group-sources?             uint32
      |       {intf-max-group-sources}?
      +--rw source-policy?
      |       -> /acl:acls/acl/name {intf-source-policy}?
      +--rw verify-source-subnet?          empty
      |       {intf-verify-source-subnet}?
      +--rw explicit-tracking?             empty
      |       {intf-explicit-tracking}?
      +--rw lite-exclude-filter?           empty
      |       {intf-lite-exclude-filter}?
      +--rw join-group*
      |       rt-types:ipv6-multicast-group-address
      |       {intf-join-group}?
      +--rw ssm-map*
      |       [ssm-map-source-addr ssm-map-group-policy]
      |       {intf-ssm-map}?
      |  +--rw ssm-map-source-addr     ssm-map-ipv6-addr-type
      |  +--rw ssm-map-group-policy    string
      +--rw static-group* [group-addr source-addr]
      |       {intf-static-group}?
      |  +--rw group-addr
      |  |       rt-types:ipv6-multicast-group-address
      |  +--rw source-addr
      |          rt-types:ipv6-multicast-source-address
```

```
            +--ro oper-status                   enumeration
            +--ro querier                       inet:ipv6-address
            +--ro joined-group*
            |       rt-types:ipv6-multicast-group-address
            |       {intf-join-group}?
            +--ro group* [group-address]
               +--ro group-address
               |       rt-types:ipv6-multicast-group-address
               +--ro expire           uint32
               +--ro filter-mode      enumeration
               +--ro up-time          uint32
               +--ro last-reporter?   inet:ipv6-address
               +--ro source* [source-address]
                  +--ro source-address    inet:ipv6-address
                  +--ro expire            uint32
                  +--ro up-time           uint32
                  +--ro host-count?       uint32
                  |       {intf-explicit-tracking}?
                  +--ro last-reporter?    inet:ipv6-address
                  +--ro host* [host-address]
                          {intf-explicit-tracking}?
                     +--ro host-address        inet:ipv6-address
                     +--ro host-filter-mode    enumeration
```

3.3. IGMP and MLD Actions

IGMP and MLD each have one action that clears the group membership cache entries for that protocol.

```
  augment /rt:routing/rt:control-plane-protocols
            /rt:control-plane-protocol:
    +--rw igmp {feature-igmp}?
       +---x clear-groups {action-clear-groups}?
          +---w input
             +---w (interface)
             |  +--:(name)
             |  |  +---w interface-name?   leafref
             |  +--:(all)
             |     +---w all-interfaces?   empty
             +---w group-address           union
             +---w source-address
                     rt-types:ipv4-multicast-source-address

  augment /rt:routing/rt:control-plane-protocols
            /rt:control-plane-protocol:
    +--rw mld {feature-mld}?
       +---x clear-groups {action-clear-groups}?
          +---w input
             +---w (interface)
             |  +--:(name)
             |  |  +---w interface-name?   leafref
             |  +--:(all)
             |     +---w all-interfaces?   empty
             +---w group-address?          union
             +---w source-address?
                     rt-types:ipv6-multicast-source-address
```

4. IGMP and MLD YANG Module

This module references [RFC1112], [RFC2236], [RFC2710], [RFC3376], [RFC3810], [RFC5790], [RFC6636], [RFC6991], [RFC8294], [RFC8343], [RFC8344], [RFC8349], and [RFC8519].

```
<CODE BEGINS> file "ietf-igmp-mld@2019-11-01.yang"
module ietf-igmp-mld {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-igmp-mld";
  prefix igmp-mld;

  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-routing-types {
  prefix rt-types;
  reference
    "RFC 8294: Common YANG Data Types for the Routing Area";
}
import ietf-access-control-list {
  prefix acl;
  reference
    "RFC 8519: YANG Data Model for Network Access Control Lists
     (ACLs)";
}
import ietf-routing {
  prefix rt;
  reference
    "RFC 8349: A YANG Data Model for Routing Management (NMDA
     Version)";
}
import ietf-interfaces {
  prefix if;
  reference
    "RFC 8343: A YANG Data Model for Interface Management";
}
import ietf-ip {
  prefix ip;
  reference
    "RFC 8344: A YANG Data Model for IP Management";
}

organization
  "IETF PIM Working Group";
contact
  "WG Web:   <http://datatracker.ietf.org/wg/pim/>
   WG List:  <mailto:pim@ietf.org>

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

   Editor:   Feng Guo
             <mailto:guofeng@huawei.com>

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

   Editor:   Pete McAllister
             <mailto:pete.mcallister@metaswitch.com>

   Editor:   Anish Peter
             <mailto:anish.ietf@gmail.com>";
description
  "The module defines the configuration and operational state for
   the Internet Group Management Protocol (IGMP) and Multicast
   Listener Discovery (MLD) protocols.

   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 (RFC 2119) (RFC 8174) when, and only when,
   they appear in all capitals, as shown here.

   Copyright (c) 2019 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 8652; see the
   RFC itself for full legal notices.";

revision 2019-11-01 {
  description
    "Initial revision.";
  reference
    "RFC 8652: A YANG Data Model for the Internet Group Management
     Protocol (IGMP) and Multicast Listener Discovery (MLD)";
}

/*
 * Features
 */

feature feature-igmp {
  description
    "Support IGMP protocol for IPv4 group membership record.";
}

feature feature-mld {
  description
    "Support MLD protocol for IPv6 group membership record.";
}

feature global-admin-enable {
  description
    "Support global configuration to enable or disable protocol.";
}

feature global-max-entries {
  description
    "Support configuration of global max-entries.";
}

feature global-max-groups {
  description
    "Support configuration of global max-groups.";
}

feature interface-global-config {
  description
    "Support global configuration applied for all interfaces.";
}

feature intf-admin-enable {
  description
    "Support configuration of interface administrative enabling.";
}

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

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

feature intf-max-groups {
  description
    "Support configuration of interface max-groups.";
}
```

```
feature intf-max-group-sources {
  description
    "Support configuration of interface max-group-sources.";
}

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

feature intf-source-policy {
  description
    "Support configuration of interface source policy.";
}

feature intf-ssm-map {
  description
    "Support configuration of interface ssm-map.";
}

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

feature intf-verify-source-subnet {
  description
    "Support configuration of interface verify-source-subnet.";
}

feature intf-explicit-tracking {
  description
    "Support configuration of interface explicit-tracking hosts.";
}

feature intf-lite-exclude-filter {
  description
    "Support configuration of interface lite-exclude-filter.";
}

feature per-interface-config {
  description
    "Support per-interface configuration.";
}

feature action-clear-groups {
  description
    "Support actions to clear groups.";
}

/*
 * Typedefs
 */

typedef ssm-map-ipv4-addr-type {
  type union {
    type enumeration {
      enum policy {
        description
          "Source address is specified in SSM map policy.";
      }
    }
    type inet:ipv4-address;
  }
  description
    "Multicast source IP address type for SSM map.";
}
// source-ipv4-addr-type

typedef ssm-map-ipv6-addr-type {
  type union {
```

```
    type enumeration {
      enum policy {
        description
          "Source address is specified in SSM map policy.";
      }
    }
    type inet:ipv6-address;
  }
  description
    "Multicast source IP address type for SSM map.";
}
// source-ipv6-addr-type

/*
 * Identities
 */

identity igmp {
  if-feature "feature-igmp";
  base rt:control-plane-protocol;
  description
    "IGMP protocol.";
  reference
    "RFC 3376: Internet Group Management Protocol, Version 3";
}

identity mld {
  if-feature "feature-mld";
  base rt:control-plane-protocol;
  description
    "MLD protocol.";
  reference
    "RFC 3810: Multicast Listener Discovery Version 2 (MLDv2) for
     IPv6";
}

/*
 * Groupings
 */

grouping global-config-attributes {
  description
    "This grouping is used in either IGMP schema or MLD schema.
     When used in IGMP schema, this grouping contains the global
     configuration for IGMP;
     when used in MLD schema, this grouping contains the global
     configuration for MLD.";
  leaf enabled {
    if-feature "global-admin-enable";
    type boolean;
    default "true";
    description
      "When this grouping is used for IGMP, this leaf indicates
       whether IGMP is enabled ('true') or disabled ('false')
       in the routing instance.
       When this grouping is used for MLD, this leaf indicates
       whether MLD is enabled ('true') or disabled ('false')
       in the routing instance.";
  }
  leaf max-entries {
    if-feature "global-max-entries";
    type uint32;
    description
      "When this grouping is used for IGMP, this leaf indicates
       the maximum number of entries in the IGMP instance.
       When this grouping is used for MLD, this leaf indicates
       the maximum number of entries in the MLD instance.
       If this leaf is not specified, the number of entries is not
       limited.";
  }
  leaf max-groups {
```

```
    if-feature "global-max-groups";
    type uint32;
    description
      "When this grouping is used for IGMP, this leaf indicates
       the maximum number of groups in the IGMP instance.
       When this grouping is used for MLD, this leaf indicates
       the maximum number of groups in the MLD instance.
       If this leaf is not specified, the number of groups is not
       limited.";
  }
}
// global-config-attributes

grouping global-state-attributes {
  description
    "This grouping is used in either IGMP schema or MLD schema.
     When used in IGMP schema, this grouping contains the global
     IGMP state attributes;
     when used in MLD schema, this grouping contains the global
     MLD state attributes.";
  leaf entries-count {
    type uint32;
    config false;
    description
      "When this grouping is used for IGMP, this leaf indicates
       the number of entries in the IGMP instance.
       When this grouping is used for MLD, this leaf indicates
       the number of entries in the MLD instance.";
  }
  leaf groups-count {
    type uint32;
    config false;
    description
      "When this grouping is used for IGMP, this leaf indicates
       the number of existing groups in the IGMP instance.
       When this grouping is used for MLD, this leaf indicates
       the number of existing groups in the MLD instance.";
  }
  container statistics {
    config false;
    description
      "When this grouping is used for IGMP, this container contains
       the statistics for the IGMP instance.
       When this grouping is used for MLD, this leaf indicates
       the statistics for the MLD instance.";
    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 error {
      description
        "Statistics of errors.";
      uses global-statistics-error;
    }
    container received {
      description
        "Statistics of received messages.";
      uses global-statistics-sent-received;
    }
    container sent {
      description
        "Statistics of sent messages.";
      uses global-statistics-sent-received;
    }
  }
```

```
  // statistics
}
// global-state-attributes

grouping global-statistics-error {
  description
    "A grouping defining statistics attributes for errors.";
  uses global-statistics-sent-received;
  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.";
  }
}
// global-statistics-error

grouping global-statistics-sent-received {
  description
    "A grouping defining statistics attributes.";
  leaf total {
    type yang:counter64;
    description
      "The number of total messages.";
  }
  leaf query {
    type yang:counter64;
    description
      "The number of query messages.";
  }
  leaf report {
    type yang:counter64;
    description
      "The number of report messages.";
  }
  leaf leave {
    type yang:counter64;
    description
      "The number of leave messages.";
  }
}
// global-statistics-sent-received

grouping interface-global-config-attributes {
  description
    "Configuration attributes applied to the interface-global level
     whose per-interface attributes are not configured.";
  leaf max-groups-per-interface {
    if-feature "intf-max-groups";
    type uint32;
    description
      "The maximum number of groups associated with each interface.
       If this leaf is not specified, the number of groups is not
       limited.";
  }
}
// interface-global-config-attributes

grouping interface-common-config-attributes {
  description
    "Configuration attributes applied to both the interface-global
     level and interface level.";
  leaf last-member-query-interval {
    type uint16 {
      range "1..1023";
    }
    units "seconds";
```

```
  description
    "When used in IGMP schema, this leaf indicates the Last
     Member Query Interval, which may be tuned to modify the
     leave latency of the network;
     when used in MLD schema, this leaf indicates the Last
     Listener Query Interval, which may be tuned to modify the
     leave latency of the network.
     This leaf is not applicable for version 1 of the IGMP.  For
     version 2 and version 3 of the IGMP, and for all versions of
     the MLD, the default value of this leaf is 1.
     This leaf may be configured at the interface level or the
     interface-global level, with precedence given to the value
     at the interface level.  If the leaf is not configured at
     either level, the default value is used.";
  reference
    "Section 8.8 of RFC 2236: Internet Group Management Protocol,
     Version 2.
     Section 8.8 of RFC 3376: Internet Group Management Protocol,
     Version 3.
     Section 7.8 of RFC 2710: Multicast Listener Discovery (MLD)
     for IPv6.
     Section 9.8 of RFC 3810: Multicast Listener Discovery
     Version 2 (MLDv2) for IPv6.";
}
leaf query-interval {
  type uint16 {
    range "1..31744";
  }
  units "seconds";
  description
    "The Query Interval is the interval between General Queries
     sent by the Querier.  In RFC 3376, the Querier's Query
     Interval (QQI) is represented from the Querier's Query
     Interval Code (QQIC) in query message as follows:
     If QQIC < 128, QQI = QQIC.
     If QQIC >= 128, QQIC represents a floating-point value as
     follows:
      0 1 2 3 4 5 6 7
     +-+-+-+-+-+-+-+-+
     |1| exp | mant  |
     +-+-+-+-+-+-+-+-+
     QQI = (mant | 0x10) << (exp + 3).
     The maximum value of QQI is 31744.
     The default value is 125.
     This leaf may be configured at the interface level or the
     interface-global level, with precedence given to the value
     at the interface level.  If the leaf is not configured at
     either level, the default value is used.";
  reference
    "Sections 4.1.7, 8.2, and 8.14.2 of RFC 3376: Internet Group
     Management Protocol, Version 3";
}
leaf query-max-response-time {
  type uint16 {
    range "1..1023";
  }
  units "seconds";
  description
    "Query maximum response time specifies the maximum time
     allowed before sending a responding report.
     The default value is 10.
     This leaf may be configured at the interface level or the
     interface-global level, with precedence given to the value
     at the interface level.  If the leaf is not configured at
     either level, the default value is used.";
  reference
    "Sections 4.1.1, 8.3, and 8.14.3 of RFC 3376: Internet Group
     Management Protocol, Version 3";
}
leaf require-router-alert {
  if-feature "intf-require-router-alert";
```

```
    type boolean;
    description
      "Protocol packets should contain the router alert IP option.
       When this leaf is not configured, the server uses the
       following rules to determine the operational value of this
       leaf:
       if this grouping is used in IGMP schema and the value of the
       leaf 'version' is 1, the value 'false' is operationally used
       by the server;
       if this grouping is used in IGMP schema and the value of the
       leaf 'version' is 2 or 3, the value 'true' is operationally
       used by the server;
       if this grouping is used in MLD schema, the value 'true' is
       operationally used by the server.
       This leaf may be configured at the interface level or the
       interface-global level, with precedence given to the value
       at the interface level.  If the leaf is not configured at
       either level, the default value is used.";
  }
  leaf robustness-variable {
    type uint8 {
      range "1..7";
    }
    description
      "The Querier's Robustness Variable allows tuning for the
       expected packet loss on a network.
       The default value is 2.
       This leaf may be configured at the interface level or the
       interface-global level, with precedence given to the value
       at the interface level.  If the leaf is not configured at
       either level, the default value is used.";
    reference
      "Sections 4.1.6, 8.1, and 8.14.1 of RFC 3376: Internet Group
       Management Protocol, Version 3";
  }
}
// interface-common-config-attributes

grouping interface-common-config-attributes-igmp {
  description
    "Configuration attributes applied to both the interface-global
     level and interface level for IGMP.";
  uses interface-common-config-attributes;
  leaf version {
    type uint8 {
      range "1..3";
    }
    description
      "IGMP version.
       The default value is 2.
       This leaf may be configured at the interface level or the
       interface-global level, with precedence given to the value
       at the interface level.  If the leaf is not configured at
       either level, the default value is used.";
    reference
      "RFC 1112: Host Extensions for IP Multicasting,
       RFC 2236: Internet Group Management Protocol, Version 2,
       RFC 3376: Internet Group Management Protocol, Version 3.";
  }
}

grouping interface-common-config-attributes-mld {
  description
    "Configuration attributes applied to both the interface-global
     level and interface level for MLD.";
  uses interface-common-config-attributes;
  leaf version {
    type uint8 {
      range "1..2";
    }
    description
```

```
      "MLD version.
       The default value is 2.
       This leaf may be configured at the interface level or the
       interface-global level, with precedence given to the value
       at the interface level.  If the leaf is not configured at
       either level, the default value is used.";
    reference
      "RFC 2710: Multicast Listener Discovery (MLD) for IPv6,
       RFC 3810: Multicast Listener Discovery Version 2 (MLDv2)
       for IPv6.";
  }
}

grouping interfaces-config-attributes-igmp {
  description
    "Configuration attributes applied to the interface-global
     level for IGMP.";
  uses interface-common-config-attributes-igmp;
  uses interface-global-config-attributes;
}

grouping interfaces-config-attributes-mld {
  description
    "Configuration attributes applied to the interface-global
     level for MLD.";
  uses interface-common-config-attributes-mld;
  uses interface-global-config-attributes;
}

grouping interface-level-config-attributes {
  description
    "This grouping is used in either IGMP schema or MLD schema.
     When used in IGMP schema, this grouping contains the IGMP
     configuration attributes that are defined at the interface
     level but are not defined at the interface-global level;
     when used in MLD schema, this grouping contains the MLD
     configuration attributes that are defined at the interface
     level but are not defined at the interface-global level.";
  leaf enabled {
    if-feature "intf-admin-enable";
    type boolean;
    default "true";
    description
      "When this grouping is used for IGMP, this leaf indicates
       whether IGMP is enabled ('true') or disabled ('false')
       on the interface.
       When this grouping is used for MLD, this leaf indicates
       whether MLD is enabled ('true') or disabled ('false')
       on the interface.";
  }
  leaf group-policy {
    type leafref {
      path "/acl:acls/acl:acl/acl:name";
    }
    description
      "When this grouping is used for IGMP, this leaf specifies
       the name of the access policy used to filter the
       IGMP membership.
       When this grouping is used for MLD, this leaf specifies
       the name of the access policy used to filter the
       MLD membership.
       The value space of this leaf is restricted to the existing
       policy instances defined by the referenced schema in
       RFC 8519.
       As specified by RFC 8519, the length of the name is between
       1 and 64; a device MAY further restrict the length of this
       name; space and special characters are not allowed.
       If this leaf is not specified, no policy is applied, and
       all packets received from this interface are accepted.";
    reference
      "RFC 8519: YANG Data Model for Network Access Control Lists
```

```
      (ACLs)";
  }
  leaf immediate-leave {
    if-feature "intf-immediate-leave";
    type empty;
    description
      "When this grouping is used for IGMP, the presence of this
       leaf requests IGMP to perform an immediate leave upon
       receiving an IGMPv2 leave message.
       If the router is IGMP-enabled, it sends an IGMP last member
       query with a last member query response time.  However, the
       router does not wait for the response time before it prunes
       the group.
       When this grouping is used for MLD, the presence of this
       leaf requests MLD to perform an immediate leave upon
       receiving an MLDv1 leave message.
       If the router is MLD-enabled, it sends an MLD last member
       query with a last member query response time.  However, the
       router does not wait for the response time before it prunes
       the group.";
  }
  leaf max-groups {
    if-feature "intf-max-groups";
    type uint32;
    description
      "When this grouping is used for IGMP, this leaf indicates
       the maximum number of groups associated with the IGMP
       interface.
       When this grouping is used for MLD, this leaf indicates
       the maximum number of groups associated with the MLD
       interface.
       If this leaf is not specified, the number of groups is not
       limited.";
  }
  leaf max-group-sources {
    if-feature "intf-max-group-sources";
    type uint32;
    description
      "The maximum number of group sources.
       If this leaf is not specified, the number of group sources
       is not limited.";
  }
  leaf source-policy {
    if-feature "intf-source-policy";
    type leafref {
      path "/acl:acls/acl:acl/acl:name";
    }
    description
      "Name of the access policy used to filter sources.
       The value space of this leaf is restricted to the existing
       policy instances defined by the referenced schema in
       RFC 8519.
       As specified by RFC 8519, the length of the name is between
       1 and 64; a device MAY further restrict the length of this
       name; space and special characters are not allowed.
       If this leaf is not specified, no policy is applied, and
       all packets received from this interface are accepted.";
  }
  leaf verify-source-subnet {
    if-feature "intf-verify-source-subnet";
    type empty;
    description
      "If present, the interface accepts packets with matching
       source IP subnet only.";
  }
  leaf explicit-tracking {
    if-feature "intf-explicit-tracking";
    type empty;
    description
      "When this grouping is used for IGMP, the presence of this
       leaf enables an IGMP-based explicit membership tracking
```

```
        function for multicast routers and IGMP proxy devices
        supporting IGMPv3.
        When this grouping is used for MLD, the presence of this
        leaf enables an MLD-based explicit membership tracking
        function for multicast routers and MLD proxy devices
        supporting MLDv2.
        The explicit membership tracking function contributes to
        saving network resources and shortening leave latency.";
      reference
        "Section 3 of RFC 6636: Tuning the Behavior of the Internet
        Group Management Protocol (IGMP) and Multicast Listener
        Discovery (MLD) for Routers in Mobile and Wireless
        Networks";
    }
    leaf lite-exclude-filter {
      if-feature "intf-lite-exclude-filter";
      type empty;
      description
        "When this grouping is used for IGMP, 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.
        When this grouping is used for MLD, 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: Lightweight Internet Group Management Protocol
        Version 3 (IGMPv3) and Multicast Listener Discovery
        Version 2 (MLDv2) Protocols";
    }
  }
  // interface-level-config-attributes

  grouping interface-config-attributes-igmp {
    description
      "Per-interface configuration attributes for IGMP.";
    uses interface-common-config-attributes-igmp;
    uses interface-level-config-attributes;
    leaf-list join-group {
      if-feature "intf-join-group";
      type rt-types:ipv4-multicast-group-address;
      description
        "The router joins this multicast group on the interface.";
    }
    list ssm-map {
      if-feature "intf-ssm-map";
      key "ssm-map-source-addr ssm-map-group-policy";
      description
        "The policy for (*,G) mapping to (S,G).";
      leaf ssm-map-source-addr {
        type ssm-map-ipv4-addr-type;
        description
          "Multicast source IPv4 address.";
      }
      leaf ssm-map-group-policy {
        type string;
        description
          "Name of the policy used to define ssm-map rules.
          A device can restrict the length
          and value of this name, possibly space and special
          characters are not allowed.";
      }
    }
    list static-group {
      if-feature "intf-static-group";
      key "group-addr source-addr";
      description
        "A static multicast route, (*,G) or (S,G).
         The version of IGMP must be 3 to support (S,G).";
      leaf group-addr {
```

```
      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.";
    }
  }
}
// interface-config-attributes-igmp

grouping interface-config-attributes-mld {
  description
    "Per-interface configuration attributes for MLD.";
  uses interface-common-config-attributes-mld;
  uses interface-level-config-attributes;
  leaf-list join-group {
    if-feature "intf-join-group";
    type rt-types:ipv6-multicast-group-address;
    description
      "The router joins this multicast group on the interface.";
  }
  list ssm-map {
    if-feature "intf-ssm-map";
    key "ssm-map-source-addr ssm-map-group-policy";
    description
      "The policy for (*,G) mapping to (S,G).";
    leaf ssm-map-source-addr {
      type ssm-map-ipv6-addr-type;
      description
        "Multicast source IPv6 address.";
    }
    leaf ssm-map-group-policy {
      type string;
      description
        "Name of the policy used to define ssm-map rules.
         A device can restrict the length
         and value of this name, possibly space and special
         characters are not allowed.";
    }
  }
  list static-group {
    if-feature "intf-static-group";
    key "group-addr source-addr";
    description
      "A static multicast route, (*,G) or (S,G).
       The version of MLD must be 2 to support (S,G).";
    leaf group-addr {
      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.";
    }
  }
}
// interface-config-attributes-mld

grouping interface-state-attributes {
  description
    "Per-interface state attributes for both IGMP and MLD.";
  leaf oper-status {
    type enumeration {
      enum up {
        description
          "Ready to pass packets.";
```

```
      }
      enum down {
        description
          "The interface does not pass any packets.";
      }
    }
    config false;
    mandatory true;
    description
      "Indicates whether the operational state of the interface
       is up or down.";
  }
}
// interface-state-attributes

grouping interface-state-attributes-igmp {
  description
    "Per-interface state attributes for IGMP.";
  uses interface-state-attributes;
  leaf querier {
    type inet:ipv4-address;
    config false;
    mandatory true;
    description
      "The querier address in the subnet.";
  }
  leaf-list joined-group {
    if-feature "intf-join-group";
    type rt-types:ipv4-multicast-group-address;
    config false;
    description
      "The routers that joined this multicast group.";
  }
  list group {
    key "group-address";
    config false;
    description
      "Multicast group membership information
       that joined on the interface.";
    leaf group-address {
      type rt-types:ipv4-multicast-group-address;
      description
        "Multicast group address.";
    }
    uses interface-state-group-attributes;
    leaf last-reporter {
      type inet:ipv4-address;
      description
        "The IPv4 address of the last host that has sent the
         report to join the multicast group.";
    }
    list source {
      key "source-address";
      description
        "List of multicast source information
         of the multicast group.";
      leaf source-address {
        type inet:ipv4-address;
        description
          "Multicast source address in group record.";
      }
      uses interface-state-source-attributes;
      leaf last-reporter {
        type inet:ipv4-address;
        description
          "The IPv4 address of the last host that has sent the
           report to join the multicast source and group.";
      }
      list host {
        if-feature "intf-explicit-tracking";
        key "host-address";
```

```
        description
          "List of hosts with the membership for the specific
           multicast source-group.";
        leaf host-address {
          type inet:ipv4-address;
          description
            "The IPv4 address of the host.";
        }
        uses interface-state-host-attributes;
      }
      // list host
    }
    // list source
  }
  // list group
}
// interface-state-attributes-igmp

grouping interface-state-attributes-mld {
  description
    "Per-interface state attributes for MLD.";
  uses interface-state-attributes;
  leaf querier {
    type inet:ipv6-address;
    config false;
    mandatory true;
    description
      "The querier address in the subnet.";
  }
  leaf-list joined-group {
    if-feature "intf-join-group";
    type rt-types:ipv6-multicast-group-address;
    config false;
    description
      "The routers that joined this multicast group.";
  }
  list group {
    key "group-address";
    config false;
    description
      "Multicast group membership information
       that joined on the interface.";
    leaf group-address {
      type rt-types:ipv6-multicast-group-address;
      description
        "Multicast group address.";
    }
    uses interface-state-group-attributes;
    leaf last-reporter {
      type inet:ipv6-address;
      description
        "The IPv6 address of the last host that has sent the
         report to join the multicast group.";
    }
    list source {
      key "source-address";
      description
        "List of multicast sources of the multicast group.";
      leaf source-address {
        type inet:ipv6-address;
        description
          "Multicast source address in group record.";
      }
      uses interface-state-source-attributes;
      leaf last-reporter {
        type inet:ipv6-address;
        description
          "The IPv6 address of the last host that has sent the
           report to join the multicast source and group.";
      }
      list host {
```

```
        if-feature "intf-explicit-tracking";
        key "host-address";
        description
          "List of hosts with the membership for the specific
           multicast source-group.";
        leaf host-address {
          type inet:ipv6-address;
          description
            "The IPv6 address of the host.";
        }
        uses interface-state-host-attributes;
      }
      // list host
    }
    // list source
  }
  // list group
}
// interface-state-attributes-mld

grouping interface-state-group-attributes {
  description
    "Per-interface state attributes for both IGMP and MLD
     groups.";
  leaf expire {
    type uint32;
    units "seconds";
    mandatory true;
    description
      "The time left before the multicast group state expires.";
  }
  leaf filter-mode {
    type enumeration {
      enum include {
        description
          "In include mode, reception of packets sent
           to the specified multicast address is requested
           only from those IP source addresses listed in the
           source-list parameter";
      }
      enum exclude {
        description
          "In exclude mode, reception of packets sent
           to the given multicast address is requested
           from all IP source addresses except those
           listed in the source-list parameter.";
      }
    }
    mandatory true;
    description
      "Filter mode for a multicast group,
       may be either include or exclude.";
  }
  leaf up-time {
    type uint32;
    units "seconds";
    mandatory true;
    description
      "The elapsed time since the device created multicast group
       record.";
  }
}
// interface-state-group-attributes

grouping interface-state-source-attributes {
  description
    "Per-interface state attributes for both IGMP and MLD
     source-group records.";
  leaf expire {
    type uint32;
    units "seconds";
```

```
    mandatory true;
    description
      "The time left before multicast source-group state expires.";
  }
  leaf up-time {
    type uint32;
    units "seconds";
    mandatory true;
    description
      "The elapsed time since the device created multicast
       source-group record.";
  }
  leaf host-count {
    if-feature "intf-explicit-tracking";
    type uint32;
    description
      "The number of host addresses.";
  }
}
// interface-state-source-attributes

grouping interface-state-host-attributes {
  description
    "Per-interface state attributes for both IGMP and MLD
     hosts of source-group records.";
  leaf host-filter-mode {
    type enumeration {
      enum include {
        description
          "In include mode.";
      }
      enum exclude {
        description
          "In exclude mode.";
      }
    }
    mandatory true;
    description
      "Filter mode for a multicast membership
       host may be either include or exclude.";
  }
}
// interface-state-host-attributes

/*
 * Configuration and Operational state data nodes (NMDA version)
 */

augment "/rt:routing/rt:control-plane-protocols/"
      + "rt:control-plane-protocol" {
  when "derived-from-or-self(rt:type, 'igmp-mld:igmp')" {
    description
      "This augmentation is only valid for a control-plane
       protocol instance of IGMP (type 'igmp').";
  }
  description
    "IGMP augmentation to routing control-plane protocol
     configuration and state.";
  container igmp {
    if-feature "feature-igmp";
    description
      "IGMP configuration and operational state data.";
    container global {
      description
        "Global attributes.";
      uses global-config-attributes;
      uses global-state-attributes;
    }
    container interfaces {
      description
        "Containing a list of interfaces.";
```

```
uses interfaces-config-attributes-igmp {
  if-feature "interface-global-config";
  refine "query-interval" {
    default "125";
  }
  refine "query-max-response-time" {
    default "10";
  }
  refine "robustness-variable" {
    default "2";
  }
  refine "version" {
    default "2";
  }
}
list interface {
  key "interface-name";
  description
    "List of IGMP interfaces.";
  leaf interface-name {
    type if:interface-ref;
    must
      '/if:interfaces/if:interface[if:name = current()]/'
    + 'ip:ipv4' {
      error-message
        "The interface must have IPv4 configured, either "
      + "enabled or disabled.";
    }
    description
      "Reference to an entry in the global interface list.";
  }
  uses interface-config-attributes-igmp {
    if-feature "per-interface-config";
    refine "last-member-query-interval" {
      must '../version != 1 or '
         + '(not(../version) and '
         + '(../../version != 1 or not(../../version)))' {
        error-message "IGMPv1 does not support "
                    + "last-member-query-interval.";
      }
    }
    refine "max-group-sources" {
      must '../version = 3 or '
         + '(not(../version) and (../../version = 3))' {
        error-message
          "The version of IGMP must be 3 to support the "
        + "source-specific parameters.";
      }
    }
    refine "source-policy" {
      must '../version = 3 or '
         + '(not(../version) and (../../version = 3))' {
        error-message
          "The version of IGMP must be 3 to support the "
        + "source-specific parameters.";
      }
    }
    refine "explicit-tracking" {
      must '../version = 3 or '
         + '(not(../version) and (../../version = 3))' {
        error-message
          "The version of IGMP must be 3 to support the "
        + "explicit tracking function.";
      }
    }
    refine "lite-exclude-filter" {
      must '../version = 3 or '
         + '(not(../version) and (../../version = 3))' {
        error-message
          "The version of IGMP must be 3 to support the "
        + "simplified EXCLUDE filter in the Lightweight "
```

```
            + "IGMPv3 protocol.";
          }
        }
      }
      uses interface-state-attributes-igmp;
    }
    // interface
  }
  // interfaces

  /*
   * Actions
   */
  action clear-groups {
    if-feature "action-clear-groups";
    description
      "Clears the specified IGMP cache entries.";
    input {
      choice interface {
        mandatory true;
        description
          "Indicates the interface(s) from which the cache
           entries are cleared.";
        case name {
          leaf interface-name {
            type leafref {
              path "/rt:routing/rt:control-plane-protocols/"
                 + "rt:control-plane-protocol/"
                 + "igmp-mld:igmp/igmp-mld:interfaces/"
                 + "igmp-mld:interface/igmp-mld:interface-name";
            }
            description
              "Name of the IGMP interface.";
          }
        }
        case all {
          leaf all-interfaces {
            type empty;
            description
              "IGMP groups from all interfaces are cleared.";
          }
        }
      }
      leaf group-address {
        type union {
          type enumeration {
            enum * {
              description
                "Any group address.";
            }
          }
          type rt-types:ipv4-multicast-group-address;
        }
        mandatory true;
        description
          "Multicast group IPv4 address.
           If the value '*' is specified, all IGMP group entries
           are cleared.";
      }
      leaf source-address {
        type rt-types:ipv4-multicast-source-address;
        mandatory true;
        description
          "Multicast source IPv4 address.
           If the value '*' is specified, all IGMP source-group
           entries are cleared.";
      }
    }
  }
  // action clear-groups
}
```

```
  // igmp
}
// augment

augment "/rt:routing/rt:control-plane-protocols/"
      + "rt:control-plane-protocol" {
  when "derived-from-or-self(rt:type, 'igmp-mld:mld')" {
    description
      "This augmentation is only valid for a control-plane
       protocol instance of IGMP (type 'mld').";
  }
  description
    "MLD augmentation to routing control-plane protocol
     configuration and state.";
  container mld {
    if-feature "feature-mld";
    description
      "MLD configuration and operational state data.";
    container global {
      description
        "Global attributes.";
      uses global-config-attributes;
      uses global-state-attributes;
    }
    container interfaces {
      description
        "Containing a list of interfaces.";
      uses interfaces-config-attributes-mld {
        if-feature "interface-global-config";
        refine "last-member-query-interval" {
          default "1";
        }
        refine "query-interval" {
          default "125";
        }
        refine "query-max-response-time" {
          default "10";
        }
        refine "require-router-alert" {
          default "true";
        }
        refine "robustness-variable" {
          default "2";
        }
        refine "version" {
          default "2";
        }
      }
      list interface {
        key "interface-name";
        description
          "List of MLD interfaces.";
        leaf interface-name {
          type if:interface-ref;
          must
            '/if:interfaces/if:interface[if:name = current()]/'
          + 'ip:ipv6' {
            error-message
              "The interface must have IPv6 configured, either "
            + "enabled or disabled.";
          }
          description
            "Reference to an entry in the global interface list.";
        }
        uses interface-config-attributes-mld {
          if-feature "per-interface-config";
          refine "max-group-sources" {
            must '../version = 2 or '
               + '(not(../version) and '
               + '(../../version = 2 or not(../../version)))' {
              error-message
```

```
            "The version of MLD must be 2 to support the "
          + "source-specific parameters.";
        }
      }
      refine "source-policy" {
        must '../version = 2 or '
           + '(not(../version) and '
           + '(../../version = 2 or not(../../version)))' {
          error-message
            "The version of MLD must be 2 to support the "
          + "source-specific parameters.";
        }
      }
      refine "explicit-tracking" {
        must '../version = 2 or '
           + '(not(../version) and '
           + '(../../version = 2 or not(../../version)))' {
          error-message
            "The version of MLD must be 2 to support the "
          + "explicit tracking function.";
        }
      }
      refine "lite-exclude-filter" {
        must '../version = 2 or '
           + '(not(../version) and '
           + '(../../version = 2 or not(../../version)))' {
          error-message
            "The version of MLD must be 2 to support the "
          + "simplified EXCLUDE filter in the Lightweight "
          + "MLDv2 protocol.";
        }
      }
    }
    uses interface-state-attributes-mld;
  }
  // interface
}
// interfaces

/*
 * Actions
 */
action clear-groups {
  if-feature "action-clear-groups";
  description
    "Clears the specified MLD cache entries.";
  input {
    choice interface {
      mandatory true;
      description
        "Indicates the interface(s) from which the cache
         entries are cleared.";
      case name {
        leaf interface-name {
          type leafref {
            path "/rt:routing/rt:control-plane-protocols/"
               + "rt:control-plane-protocol/"
               + "igmp-mld:mld/igmp-mld:interfaces/"
               + "igmp-mld:interface/igmp-mld:interface-name";
          }
          description
            "Name of the MLD interface.";
        }
      }
      case all {
        leaf all-interfaces {
          type empty;
          description
            "MLD groups from all interfaces are cleared.";
        }
      }
```

```
            }
            leaf group-address {
              type union {
                type enumeration {
                  enum * {
                    description
                      "Any group address.";
                  }
                }
                type rt-types:ipv6-multicast-group-address;
              }
              description
                "Multicast group IPv6 address.
                 If the value '*' is specified, all MLD group entries
                 are cleared.";
            }
            leaf source-address {
              type rt-types:ipv6-multicast-source-address;
              description
                "Multicast source IPv6 address.
                 If the value '*' is specified, all MLD source-group
                 entries are cleared.";
            }
          }
        }
        // action clear-mld-groups
      }
      // mld
    }
    // augment
  }
  <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/igmp-mld:igmp,

igmp-mld:global
: This subtree specifies the configuration for the IGMP attributes at the global level on an IGMP instance. Modifying the configuration can cause IGMP membership to be deleted or reconstructed on all the interfaces of an IGMP instance.

igmp-mld:interfaces
: This subtree specifies the configuration for the IGMP attributes at the interface-global level on an IGMP instance. Modifying the configuration can cause IGMP membership to be deleted or reconstructed on all the interfaces of an IGMP instance.

igmp-mld:interfaces/interface
   This subtree specifies the configuration for the IGMP attributes at the interface level on an IGMP instance. Modifying the configuration can cause IGMP membership to be deleted or reconstructed on a specific interface of an IGMP instance.

Under /rt:routing/rt:control-plane-protocols
/rt:control-plane-protocol/igmp-mld:mld,

igmp-mld:global
   This subtree specifies the configuration for the MLD attributes at the global level on an MLD instance. Modifying the configuration can cause MLD membership to be deleted or reconstructed on all the interfaces of an MLD instance.

igmp-mld:interfaces
   This subtree specifies the configuration for the MLD attributes at the interface-global level on an MLD instance. Modifying the configuration can cause MLD membership to be deleted or reconstructed on all the interfaces of an MLD instance.

igmp-mld:interfaces/interface
   This subtree specifies the configuration for the MLD attributes at the interface level on a device. Modifying the configuration can cause MLD membership to be deleted or reconstructed on a specific interface of an MLD instance.

Unauthorized access to any data node of these subtrees can adversely affect the membership records of multicast routing subsystem on the local device. 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:

/rt:routing/rt:control-plane-protocols
/rt:control-plane-protocol/igmmp-mld:igmp

/rt:routing/rt:control-plane-protocols
/rt:control-plane-protocol/igmp-mld:mld

Unauthorized access to any data node of the above subtree can disclose the operational state information of IGMP or MLD on this device.

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:

/rt:routing/rt:control-plane-protocols
/rt:control-plane-protocol/igmmp-mld:igmp/igmmp-mld:clear-groups

/rt:routing/rt:control-plane-protocols
/rt:control-plane-protocol/igmp-mld:mld/igmp-mld:clear-groups

Unauthorized access to any of the above action operations can delete the IGMP or MLD membership records on this device.

6. IANA Considerations

This document registers the following namespace URIs in the IETF XML registry [RFC3688]:

URI: urn:ietf:params:xml:ns:yang:ietf-igmp-mld

Registrant Contact: The IESG.

XML: N/A; the requested URI is an XML namespace.

This document registers the following YANG modules in the YANG Module Names registry [RFC6020]:

Name: ietf-igmp-mld

Namespace: urn:ietf:params:xml:ns:yang:ietf-igmp-mld

Prefix: igmp-mld

Reference: RFC 8652

7. References

7.1. Normative References

[RFC1112] Deering, S., "Host extensions for IP multicasting", STD 5, RFC 1112, DOI 10.17487/RFC1112, August 1989, <https://www.rfc-editor.org/info/rfc1112>.

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

[RFC2236] Fenner, W., "Internet Group Management Protocol, Version 2", RFC 2236, DOI 10.17487/RFC2236, November 1997, <https://www.rfc-editor.org/info/rfc2236>.

[RFC2710] Deering, S., Fenner, W., and B. Haberman, "Multicast Listener Discovery (MLD) for IPv6", RFC 2710, DOI 10.17487/RFC2710, October 1999, <https://www.rfc-editor.org/info/rfc2710>.

[RFC3376] Cain, B., Deering, S., Kouvelas, I., Fenner, B., and A. Thyagarajan, "Internet Group Management Protocol, Version 3", RFC 3376, DOI 10.17487/RFC3376, October 2002, <https://www.rfc-editor.org/info/rfc3376>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

[RFC3810] Vida, R., Ed. and L. Costa, Ed., "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", RFC 3810, DOI 10.17487/RFC3810, June 2004, <https://www.rfc-editor.org/info/rfc3810>.

[RFC4607] Holbrook, H. and B. Cain, "Source-Specific Multicast for IP", RFC 4607, DOI 10.17487/RFC4607, August 2006, <https://www.rfc-editor.org/info/rfc4607>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <https://www.rfc-editor.org/info/rfc6242>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/info/rfc7950>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8294] Liu, X., Qu, Y., Lindem, A., Hopps, C., and L. Berger, "Common YANG Data Types for the Routing Area", RFC 8294, DOI 10.17487/RFC8294, December 2017, <https://www.rfc-editor.org/info/rfc8294>.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <https://www.rfc-editor.org/info/rfc8341>.

[RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <https://www.rfc-editor.org/info/rfc8342>.

[RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, DOI 10.17487/RFC8343, March 2018, <https://www.rfc-editor.org/info/rfc8343>.

[RFC8344] Bjorklund, M., "A YANG Data Model for IP Management", RFC 8344, DOI 10.17487/RFC8344, March 2018, <https://www.rfc-editor.org/info/rfc8344>.

[RFC8349] Lhotka, L., Lindem, A., and Y. Qu, "A YANG Data Model for Routing Management (NMDA Version)", RFC 8349, DOI 10.17487/RFC8349, March 2018, <https://www.rfc-editor.org/info/rfc8349>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC8519] Jethanandani, M., Agarwal, S., Huang, L., and D. Blair, "YANG Data Model for Network Access Control Lists (ACLs)", RFC 8519, DOI 10.17487/RFC8519, March 2019, <https://www.rfc-editor.org/info/rfc8519>.

7.2. Informative References

[RFC3569] Bhattacharyya, S., Ed., "An Overview of Source-Specific Multicast (SSM)", RFC 3569, DOI 10.17487/RFC3569, July 2003, <https://www.rfc-editor.org/info/rfc3569>.

[RFC4541] Christensen, M., Kimball, K., and F. Solensky, "Considerations for Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD) Snooping Switches", RFC 4541, DOI 10.17487/RFC4541, May 2006, <https://www.rfc-editor.org/info/rfc4541>.

[RFC4605] Fenner, B., He, H., Haberman, B., and H. Sandick, "Internet Group Management Protocol (IGMP) / Multicast Listener Discovery (MLD)-Based Multicast Forwarding ("IGMP/MLD Proxying")", RFC 4605, DOI 10.17487/RFC4605, August 2006, <https://www.rfc-editor.org/info/rfc4605>.

[RFC5790] Liu, H., Cao, W., and H. Asaeda, "Lightweight Internet Group Management Protocol Version 3 (IGMPv3) and Multicast Listener Discovery Version 2 (MLDv2) Protocols", RFC 5790, DOI 10.17487/RFC5790, February 2010,

<https://www.rfc-editor.org/info/rfc5790>.

[RFC6636] Asaeda, H., Liu, H., and 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, DOI 10.17487/RFC6636, May 2012, <https://www.rfc-editor.org/info/rfc6636>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

[RFC8407] Bierman, A., "Guidelines for Authors and Reviewers of Documents Containing YANG Data Models", BCP 216, RFC 8407, DOI 10.17487/RFC8407, October 2018, <https://www.rfc-editor.org/info/rfc8407>.

[RFC8639] Voit, E., Clemm, A., Gonzalez Prieto, A., Nilsen-Nygaard, E., and A. Tripathy, "Subscription to YANG Notifications", RFC 8639, DOI 10.17487/RFC8639, September 2019, <https://www.rfc-editor.org/info/rfc8639>.

[RFC8641] Clemm, A. and E. Voit, "Subscription to YANG Notifications for Datastore Updates", RFC 8641, DOI 10.17487/RFC8641, September 2019, <https://www.rfc-editor.org/info/rfc8641>.

Acknowledgments

The authors would like to thank Steve Baillargeon, Hu Fangwei, Robert Kebler, Tanmoy Kundu, and Stig Venaas for their valuable contributions.

Contributors

Yisong Liu
Huawei Technologies
China

Email: liuyisong@huawei.com

Authors' Addresses

Xufeng Liu
Volta Networks

Email: xufeng.liu.ietf@gmail.com

Feng Guo
China
100095
Beijing
Huawei Bldg., No. 156 Beiqing Rd., Haidian District
Huawei Technologies

Email: guofeng@huawei.com

Mahesh Sivakumar
Juniper Networks
1133 Innovation Way
Sunnyvale, California
United States of America

Email: sivakumar.mahesh@gmail.com

Pete McAllister
Metaswitch Networks
100 Church Street
Enfield

EN2 6BQ
United Kingdom

Email: pete.mcallister@metaswitch.com

Anish Peter
IP Infusion India
RMZ Centennial, Block D 401
Kundanahalli Main Road, Mahadevapura Post
Bangalore
India

Email: anish.ietf@gmail.com