PIM WG
Internet-Draft
Intended status: Standards Track
Expires: April 15, 2019

Xufeng. Liu
Volta Networks
Zheng. Zhang
ZTE Corporation
Anish. Peter
Individual contributor
Mahesh. Sivakumar
Juniper networks
Feng. Guo
Huawei Technologies
Pete. McAllister
Metaswitch Networks
October 12, 2018

A YANG Data Model for Multicast Source Discovery Protocol (MSDP) draft-ietf-pim-msdp-yang-05

Abstract

This document defines a YANG data model for the configuration and management of Multicast Source Discovery Protocol (MSDP) Protocol.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 15, 2019.

Table of Contents

1. Introduction

[RFC3618] introduces the protocol definition of MSDP. This document defines a YANG data model that can be used to configure and manage the MSDP protocol. The operational state data and statistics can also be retrieved by this model.

This model is designed to be used along with other multicast YANG models such as PIM, which are not covered in this document.

2. Design of the Data Model

This model imports and augments ietf-routing YANG model defined in [RFC8349]. Both configuration data nodes and state data nodes of [RFC8349] are augmented. The configuration data nodes cover global configuration attributes and per peer configuration attributes. The state data nodes include global, per peer, and source-active information. The container "msdp" is the top level container in this data model. The presence of this container is expected to enable MSDP protocol functionality. No notification is defined in this model.

```
module: ietf-msdp
  augment /rt:routing/rt:control-plane-protocols:
    +--rw msdp!
       +--rw global
```

```
       |  +--rw tcp-connection-source?   if:interface-ref
       |  +--rw default-peer* [peer-addr prefix-policy] {global-default-
peer,global-default-peer-policy}?
       |  |  +--rw peer-addr        -> ../../../peers/peer/address
       |  |  +--rw prefix-policy    string
       |  +--rw originating-rp
       |  |  +--rw interface?   if:interface-ref
       |  +--rw sa-filter
       |  |  +--rw in?    string
       |  |  +--rw out?   string
       |  +--rw sa-limit?               uint32 {global-sa-limit}?
       |  +--rw ttl-threshold?          uint8
       +--rw peers
       |  +--rw peer* [address]
       |     +--rw address                  inet:ipv4-address
       |     +--rw authentication
       |     |  +--rw (authentication-type)?
       |     |     +--:(key-chain) {peer-key-chain}?
       |     |     |  +--rw key-chain?          key-chain:key-chain-ref
       |     |     +--:(password)
       |     |        +--rw key?                string
       |     |        +--rw crypto-algorithm?   identityref
       |     +--rw enable?                  boolean {peer-admin-enable}?
       |     +--rw tcp-connection-source?   if:interface-ref
       |     +--rw description?             string {peer-description}?
       |     +--rw mesh-group?              string
       |     +--rw peer-as?                 inet:as-number {peer-as}?
       |     +--rw sa-filter
       |     |  +--rw in?    string
       |     |  +--rw out?   string
       |     +--rw sa-limit?                uint32 {peer-sa-limit}?
       |     +--rw timer
       |     |  +--rw connect-retry-interval?   uint16
       |     |  +--rw holdtime-interval?        uint16
       |     |  +--rw keepalive-interval?       uint16
       |     +--rw ttl-threshold?           uint8
       |     +--ro session-state?           enumeration
       |     +--ro elapsed-time?            uint32
       |     +--ro connect-retry-expire?    uint32
       |     +--ro hold-expire?             uint16
       |     +--ro is-default-peer?         boolean
       |     +--ro keepalive-expire?        uint16
       |     +--ro reset-count?             uint32
       |     +--ro statistics
       |        +--ro discontinuity-time?   yang:date-and-time
       |        +--ro error
       |        |  +--ro rpf-failure?   uint32
       |        +--ro queue
```

```
       |        |  +--ro size-in?    uint32
```

```
      |        |  +--ro size-out?   uint32
      |        +--ro received
      |        |  +--ro keepalive?      yang:counter64
      |        |  +--ro notification?   yang:counter64
      |        |  +--ro sa-message?     yang:counter64
      |        |  +--ro sa-response?    yang:counter64
      |        |  +--ro sa-request?     yang:counter64
      |        |  +--ro total?          yang:counter64
      |        +--ro sent
      |           +--ro keepalive?      yang:counter64
      |           +--ro notification?   yang:counter64
      |           +--ro sa-message?     yang:counter64
      |           +--ro sa-response?    yang:counter64
      |           +--ro sa-request?     yang:counter64
      |           +--ro total?          yang:counter64
      +--ro sa-cache
         +--ro entry* [group source-addr]
            +--ro group               inet:ipv4-address
            +--ro source-addr         union
            +--ro origin-rp* [rp-address]
            |  +--ro rp-address       inet:ip-address
            |  +--ro is-local-rp?     boolean
            |  +--ro sa-adv-expire?   uint32
            +--ro state-attributes
               +--ro up-time?             uint32
               +--ro expire?              uint32
               +--ro holddown-interval?   uint32
               +--ro peer-learned-from?   inet:ipv4-address
               +--ro rpf-peer?            inet:ipv4-address

  rpcs:
    +---x clear-peer
    |  +---w input
    |     +---w peer-address?   inet:ipv4-address
    +---x clear-sa-cache {rpc-clear-sa-cache}?
       +---w input
          +---w entry!
          |  +---w group          rt-types:ipv4-multicast-group-address
          |  +---w source-addr?   rt-types:ipv4-multicast-source-address
          +---w peer-address?   inet:ipv4-address
          +---w peer-as?        inet:as-number
```

3. MSDP configuration

MSDP configurations require peer configurations. Several peers may be configured in a mesh-group. The Source-Active information may be filtered by peers.

The configuration modeling branch is composed of MSDP global and peer configurations. The two parts are the most important parts of MSDP.

Besides the fundamental features of MSDP protocol, several optional features are included in the model. These features help the control of MSDP protocol. The peer features and SA features make the deployment and control easier. The connection parameters can be used to control the TCP connection because MSDP protocol is based on TCP. The authentication features make the protocol more secure. The filter features selectively allow operators to prevent SA information from being forwarded to peers.

4. MSDP State

MSDP states are composed of MSDP global state, MSDP peer state, statistics information and SA cache information. The statistics information and SA cache information helps the operator to retrieve the protocol condition.

5. MSDP RPC

The RPC part is used to define some useful and ordinary operations of protocol management. Network manager can delete all the information from a given peer by using the clear-peer rpc. And network manager can delete a given SA cache information by clear-sa-cache rpc.

6. MSDP YANG model

```
<CODE BEGINS> file "ietf-msdp.yang"
   module ietf-msdp {

    yang-version 1.1;

     namespace "urn:ietf:params:xml:ns:yang:ietf-msdp";
     prefix msdp;

     import ietf-yang-types {
       prefix "yang";
       reference "RFC6991";
     }

     import ietf-inet-types {
       prefix "inet";
       reference "RFC6991";
     }

     import ietf-routing {
       prefix "rt";
```

```
      reference "RFC8022";
    }

    import ietf-interfaces {
      prefix "if";
      reference "RFC7223";
    }

    import ietf-ip {
      prefix "ip";
      reference "RFC7277";
    }

    import ietf-key-chain {
      prefix "key-chain";
      reference "RFC8177";
    }

    import ietf-routing-types {
      prefix "rt-types";
      reference "RFC8294";
    }

    organization
      "IETF PIM(Protocols for IP Multicast) Working Group";

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

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

       Editor:   Zheng Zhang
                 <mailto:zzhang_ietf@hotmail.com>

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

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

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

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

```
        description
         "The module defines the YANG definitions for MSDP.

          Copyright (c) 2018 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 3618; see
          the RFC itself for full legal notices.";

      revision 2018-10-20 {
        description
          "Initial revision.";
        reference
          "RFC XXXX: A YANG Data Model for MSDP.
           RFC 3618: Multicast Source Discovery Protocol (MSDP).
           RFC 4624: Multicast Source Discovery Protocol (MSDP) MIB";
      }

      /*
       * Features
       */
      feature global-tcp-connect-source {
        description
          "Support configuration of global tcp connect source.";
      }

      feature global-default-peer {
        description
          "Support configuration of global default peer.";
      }

      feature global-default-peer-policy {
        description
          "Support policy configuration of global default peer.";
      }

      feature global-sa-filter {
        description
          "Support configuration of global SA filter.";
      }

      feature global-sa-limit {
```

```
      description
        "Support configuration of global limit on SA entries.";
    }

    feature global-ttl-threshold {
      description
        "Support configuration of global TTL threshold.";
    }

    feature rpc-clear-sa-cache {
      description
        "Support the RPC to clear SA cache.";
    }

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

    feature peer-as {
      description
        "Support configuration of peer AS number.";
    }

    feature peer-tcp-connect-source {
      description
        "Support configuration of peer tcp connect source.";
    }

    feature peer-description {
      description
        "Support configuration of peer description.";
    }

    feature peer-key-chain {
      description
        "Support configuration of peer key-chain.";
    }

    feature peer-password {
      description
        "Support configuration of peer password.";
    }

    feature peer-sa-limit {
      description
        "Support configuration of per peer limit on SA entries.";
    }
```

```
    /*
     * Groupings
     */
    grouping authentication-container {
      description
        "Authentication attributes.";
      container authentication {
        description
          "A container defining authentication attributes.";
        choice authentication-type {
          case key-chain {
            if-feature peer-key-chain;
            leaf key-chain {
              type key-chain:key-chain-ref;
              description
                "Reference to a key-chain.";
            }
          }
          case password {
            leaf key {
              type string;
              description
                "This leaf describes the authentication key.";
            }
            leaf crypto-algorithm {
              type identityref {
                base key-chain:crypto-algorithm;
              }
              description
                "Cryptographic algorithm associated with key.";
            }
          }
          description
            "Choice of authentication.";
        }
      }
    } // authentication-container

    grouping tcp-connect-source {
      description
        "Attribute to configure peer TCP connection source.";
      leaf tcp-connection-source {
        type if:interface-ref;
        must "/if:interfaces/if:interface[if:name = current()]/"
          + "ip:ipv4" {
          description
            "The interface must have IPv4 enabled.";
        }
```

```
        description
          "The interface is to be the source for the TCP
           connection. It is a reference to an entry in the global
           interface list.";
      }
    } // tcp-connection-source

    grouping global-config-attributes {
      description "Global MSDP configuration.";

      uses tcp-connect-source {
        if-feature global-tcp-connect-source;
      }
      list default-peer {
        if-feature global-default-peer;
        if-feature global-default-peer-policy;
        key "peer-addr prefix-policy";

        description
          "The default peer accepts all MSDP SA messages.
           A default peer is needed in topologies where MSDP peers
           do not coexist with BGP peers. The reverse path
           forwarding (RPF) check on SA messages can fail, and no
           SA messages are accepted. In these cases, you can configure
           the peer as a default peer and bypass RPF checks.";

        leaf peer-addr {
          type leafref {
            path "../../../peers/peer/address";
          }
          mandatory true;
          description
            "Reference to a peer that is in the peer list.";
        }
        leaf prefix-policy {
          type string;
          description
            "If specified, only those SA entries whose RP is
             permitted in the prefix list are allowed;
             if not specified, all SA messages from the default
             peer are accepted.";
        }
      } // default-peer

      container originating-rp {
        description
          "The container of Originating RP.";
        leaf interface {
```

```
          type if:interface-ref;
          must "/if:interfaces/if:interface[if:name = current()]/"
            + "ip:ipv4" {
            description
              "The interface must have IPv4 enabled.";
          }
          description
            "Reference to an entry in the global interface
             list.
             IP address of the interface is used in the RP field of
             an SA message entry. When Anycast RPs are used, all
             RPs use the same IP address. This parameter can be
             used to define a unique IP address for the RP of each
             MSDP peer.
             By default, the software uses the RP address of the
             local system.";
        }
      } // originating-rp

      uses sa-filter-container {
        if-feature global-sa-filter;
      }
      leaf sa-limit {
        if-feature global-sa-limit;
        type uint32;
        description
          "A limit on the number of SA entries accepted.
           By default, there is no limit.";
      }
      uses ttl-threshold {
        if-feature global-ttl-threshold;
      }
    } // global-config-attributes

    grouping peer-config-attributes {
      description "Per peer configuration for MSDP.";

      uses authentication-container;
      leaf enable {
        if-feature peer-admin-enable;
        type boolean;
        description
          "'true' if peer is enabled;
           'false' if peer is disabled.";
      }
      uses tcp-connect-source {
        if-feature peer-tcp-connect-source;
      }
```

```
        leaf description {
          if-feature peer-description;
          type string;
          description
            "The peer description.";
        }
        leaf mesh-group {
          type string;
          description
            "Configure this peer to be a member of a mesh group";
        }
        leaf peer-as {
          if-feature peer-as;
          type inet:as-number;
          description
            "Peer's autonomous system number (ASN). Using peer-as to
             do verification can provide more controlled ability.";
        }
        uses sa-filter-container;
        leaf sa-limit {
          if-feature peer-sa-limit;
          type uint32;
          description
            "A limit on the number of SA entries accepted from this
             peer. By default, there is no limit.";
        }
        container timer {
          description "Timer attributes.";
          leaf connect-retry-interval {
            type uint16;
            units seconds;
            default 30;
            description "Peer timer for connect-retry,
                         SHOULD be set to 30 seconds.";
          }
          leaf holdtime-interval {
            type uint16 {
                range "3..65535";
            }
            units seconds;
            must "(../keepalive-interval and . > ../keepalive-interval) or "
                   +"(not(../keepalive-interval) and . > 60)" {
              error-message "The keep alive interval must be "
                          + "smaller than the hold time interval";
            }
            default 75;
            description "The SA hold down period of this MSDP peer.";
          }
```

```
        leaf keepalive-interval {
          type uint16 {
              range "1..65535";
          }
          units seconds;
          must "(../holdtime-interval and . < ../holdtime-interval) or "
                      +"(not(../holdtime-interval) and . < 75)" {
            error-message "The keep alive interval must be "
                      + "smaller than the hold time interval";
          }
          default 60;
          description "The keepalive timer of this MSDP peer.";
        }
      } // timer
      uses ttl-threshold;
    } // peer-config-attributes

    grouping peer-state-attributes {
      description "Per peer state attributes for MSDP.";

      leaf session-state {
        type enumeration {
          enum disabled  {
            description "Disabled.";
          }
          enum inactive {
            description "Inactive.";
          }
          enum listen {
            description "Listen.";
          }
          enum connecting {
            description "Connecting.";
          }
          enum established {
            description "Established.";
          }
        }
        config false;
        description
          "Peer session state.";
        reference
          "RFC3618: Multicast Source Discovery Protocol (MSDP).";
      }
      leaf elapsed-time {
        type uint32;
        units seconds;
        config false;
```

```
        description "Elapsed time for being in a state.";
      }
      leaf connect-retry-expire {
        type uint32;
        units seconds;
        config false;
        description "Connect retry expire time of peer connection.";
      }
      leaf hold-expire {
        type uint16;
        units seconds;
        config false;
        description "Hold expire time of peer connection.";
      }
      leaf is-default-peer {
        type boolean;
        config false;
        description "If this peer is default peer.";
      }
      leaf keepalive-expire {
        type uint16;
        units seconds;
        config false;
        description "Keepalive expire time of this peer.";
      }
      leaf reset-count {
        type uint32;
        config false;
        description "The reset count of this peer.";
      }

      container statistics {
        config false;
        description
          "A container defining statistics attributes.";

        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
            "A grouping defining error statistics attributes.";
          leaf rpf-failure {
            type uint32;
            description "Number of RPF failures.";
          }
        } // statistics-error

        container queue {
          description
            "A container includes queue statistics attributes.";
          leaf size-in {
            type uint32;
            description
              "The size of the input queue.";
          }
          leaf size-out {
            type uint32;
            description
              "The size of the output queue.";
          }
        } // statistics-queue

        container received {
          description "Received message counters.";
          uses statistics-sent-received;
        }
        container sent {
          description "Sent message counters.";
          uses statistics-sent-received;
        }
      } // statistics-container
    } // peer-state-attributes

    grouping sa-filter-container {
      description "A container defining SA filters.";
      container sa-filter {
        description
          "Specifies an access control list (ACL) to filter source
           active (SA) messages coming in to or going out of the
           peer.";
        leaf in {
          type string;
          description
            "Filters incoming SA messages only.
             The string value is the name to uniquely identify a policy that
             contains one or more policy rules used to accept or reject MSDP
             SA messages.
```

```
              If a policy is not specified, all MSDP SA messages are accepted,
              The definition of such a policy is outside the scope of this
document.";
          }
          leaf out {
            type string;
            description
              "Filters outgoing SA messages only.
               The string value is the name to uniquely identify a policy that
               contains one or more policy rules used to accept or reject MSDP
               SA messages.
               If a policy is not specified, all MSDP SA messages are accepted,
               The definition of such a policy is outside the scope of this
document.";
          }
        } // sa-filter
      } // sa-filter-container

      grouping ttl-threshold {
        description "Attribute to configure TTL threshold.";
        leaf ttl-threshold {
          type uint8 {
            range 1..255;
          }
          description "Maximum number of hops data packets can
                       traverse before being dropped.";
        }
      } // sa-ttl-threshold

      grouping statistics-sent-received {
        description
          "A grouping defining sent and received statistics attributes.";
        leaf keepalive {
          type yang:counter64;
          description
            "The number of keepalive messages.";
        }
        leaf notification {
          type yang:counter64;
          description
            "The number of notification messages.";
        }
        leaf sa-message {
          type yang:counter64;
          description
            "The number of SA messages.";
        }
        leaf sa-response {
```

```
          type yang:counter64;
          description
```

```
          "The number of SA response messages.";
      }
      leaf sa-request {
        type yang:counter64;
        description
          "The number of SA request messages.";
      }
      leaf total {
        type yang:counter64;
        description
          "The number of total messages.";
      }
    } // statistics-sent-received

    /*
     * Data nodes
     */
    augment "/rt:routing/rt:control-plane-protocols" {
      description
        "MSDP augmentation to routing instance. This augmentation
         is only valid for a routing protocol instance of MSDP.";

      container msdp {
        presence "Container for MSDP protocol.";
        description
          "MSDP configuration data.";

        container global {
          description
            "Global attributes.";
          uses global-config-attributes;
        }

        container peers {
          description
            "Containing a list of peers.";
          list peer {
            key "address";
            description
              "List of MSDP peers.";
            leaf address {
              type inet:ipv4-address;
              description
                "The address of peer";
            }
            uses peer-config-attributes;
            uses peer-state-attributes;
          } // peer
```

```
        } // peers

        container sa-cache {
          config false;
          description
            "The SA cache information.";
          list entry {
            key "group source-addr";
            description "A list of SA cache entries.";
            leaf group {
              type inet:ipv4-address;
              description "The group address of this SA cache.";
            }
            leaf source-addr {
              type union {
                type enumeration {
                  enum '*' {
                    description "Any source address.";
                  }
                }
                type inet:ipv4-address;
              }
              description "Source IPv4 address.";
            }
            list origin-rp {
              key "rp-address";
              description "Origin RP address.";
              leaf rp-address {
                type inet:ip-address;
                description "The RP address.";
              }
              leaf is-local-rp {
                type boolean;
                description "The RP is local.";
              }
              leaf sa-adv-expire {
                type uint32;
                units seconds;
                description
                  "The remaining time duration before expiration
                   of the periodic SA advertisement timer on a local RP.";
              }
            }

            container state-attributes {
              description "SA cache state attributes for MSDP.";

              leaf up-time {
```

```
                type uint32;
                units seconds;
                description "The duration time of receiving this SA cache.";
              }
              leaf expire {
                type uint32;
                units seconds;
                description "The duration time since this SA cache expires.";
              }
              leaf holddown-interval {
                type uint32;
                units seconds;
                description "Hold-down timer value for SA forwarding.";
              }
              leaf peer-learned-from {
                type inet:ipv4-address;
                description
                  "The address of the peer that we learned this SA from.";
              }
              leaf rpf-peer {
                type inet:ipv4-address;
                description
                  "The address is used to find the SA's originating RP.";
              }
            } // sa-cache-state-attributes
          } // entry
        } // sa-cache
      } // msdp
    } // augment

    /*
     * RPCs
     */
    rpc clear-peer {
      description
        "Clears the TCP connection to the peer.";
      input {
        leaf peer-address {
          type inet:ipv4-address;
          description
            "Address of peer to be cleared. If this is not provided
             then all peers are cleared.";
        }
      }
    }

    rpc clear-sa-cache {
      if-feature rpc-clear-sa-cache;
```

```
       description
         "Clears MSDP source active (SA) cache entries.";
       input {
         container entry {
           presence "If a particular entry is cleared.";
           description
             "The SA cache (S,G) or (*,G) entry to be cleared. If
              this is not provided, all entries are cleared.";
           leaf group {
             type rt-types:ipv4-multicast-group-address;
             mandatory true;
             description "The group address";
           }
           leaf source-addr {
              type rt-types:ipv4-multicast-source-address;
              description
                "Address of multicast source to be cleared. If this
                 is not provided then all entries related to the
                 given group are cleared.";
           }
         } // s-g
         leaf peer-address {
           type inet:ipv4-address;
           description
             "Peer IP address from which MSDP SA cache entries have
              been learned. If this is not provided, entries learned
              from all peers are cleared.";
         }
         leaf peer-as {
           type inet:as-number;
           description
             "ASN from which MSDP SA cache entries have been learned.
              If this is not provided, entries learned from all AS's
              are cleared.";
         }
       }
     }
   }
<CODE ENDS>
```

7. 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 [RFC5246].

The NETCONF access control model [RFC6536] 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. For MSDP, the ability to modify MSDP configuration will allow the unexpected MSDP peer establishment and unexpected SA information learning and advertisement. The "password" field is also a sensitive readable configuration, the unauthorized reading function may lead to the password leaking. The security considerations of MSDP [RFC3618] are applicable.

The RPC 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. The MSDP Yang module supports the "clear-peer" and "clear-sa-cache" RPCs. If access to either of these is compromised, they can result in unexpected MSDP peer breakdown and unexpected SA information deletion.

8. IANA Considerations

The IANA is requested to assign two new URIs from the IETF XML registry ([RFC3688]). Authors are suggesting the following URI:

URI: urn:ietf:params:xml:ns:yang:ietf-msdp

Registrant Contact: PIM WG

XML: N/A, the requested URI is an XML namespace

This document also requests one new YANG module name in the YANG Module Names registry ([RFC6020]) with the following suggestion:

name: ietf-msdp

namespace: urn:ietf:params:xml:ns:yang:ietf-msdp

prefix: msdp

reference: RFC XXXX

9. Contributors

The authors would like to thank Yisong Liu (liuyisong@huawei.com), Benchong Xu (xu.benchong@zte.com.cn), Tanmoy Kundu (tanmoy.kundu@alcatel-lucent.com) for their valuable contributions.

10. Acknowledgement

The authors would like to thank Stig Venaas, Jake Holland for their valuable comments and suggestions.

11. Normative References

[I-D.ietf-netmod-rfc6087bis]
Bierman, A., "Guidelines for Authors and Reviewers of YANG Data Model Documents", draft-ietf-netmod-rfc6087bis-20 (work in progress), March 2018.

[I-D.ietf-rtgwg-policy-model]
Qu, Y., Tantsura, J., Lindem, A., Liu, X., and A. Shaikh, "A YANG Data Model for Routing Policy Management", draft-ietf-rtgwg-policy-model-03 (work in progress), June 2018.

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

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

[RFC4624] Fenner, B. and D. Thaler, "Multicast Source Discovery Protocol (MSDP) MIB", RFC 4624, DOI 10.17487/RFC4624, October 2006, <https://www.rfc-editor.org/info/rfc4624>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <https://www.rfc-editor.org/info/rfc5246>.

[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>.

[RFC6087] Bierman, A., "Guidelines for Authors and Reviewers of YANG Data Model Documents", RFC 6087, DOI 10.17487/RFC6087, January 2011, <https://www.rfc-editor.org/info/rfc6087>.

[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>.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <https://www.rfc-editor.org/info/rfc6536>.

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

[RFC7223] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 7223, DOI 10.17487/RFC7223, May 2014, <https://www.rfc-editor.org/info/rfc7223>.

[RFC7277] Bjorklund, M., "A YANG Data Model for IP Management", RFC 7277, DOI 10.17487/RFC7277, June 2014, <https://www.rfc-editor.org/info/rfc7277>.

[RFC8022] Lhotka, L. and A. Lindem, "A YANG Data Model for Routing Management", RFC 8022, DOI 10.17487/RFC8022, November 2016, <https://www.rfc-editor.org/info/rfc8022>.

[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>.

[RFC8177] Lindem, A., Ed., Qu, Y., Yeung, D., Chen, I., and J. Zhang, "YANG Data Model for Key Chains", RFC 8177, DOI 10.17487/RFC8177, June 2017, <https://www.rfc-editor.org/info/rfc8177>.

[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>.

Authors' Addresses

Xufeng Liu
Volta Networks

Email: xufeng.liu.ietf@gmail.com

Zheng Zhang
ZTE Corporation
No. 50 Software Ave, Yuhuatai Distinct
Nanjing
China

Email: zzhang_ietf@hotmail.com

Anish Peter
Individual contributor

Email: anish.ietf@gmail.com

Mahesh Sivakumar
Juniper networks
1133 Innovation Way
Sunnyvale, CALIFORNIA 94089
USA

Email: sivakumar.mahesh@gmail.com

Feng Guo
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: guofeng@huawei.com

Pete McAllister
Metaswitch Networks
100 Church Street
Enfield EN2 6BQ
UK

Email: pete.mcallister@metaswitch.com