PIM WG
Internet-Draft
Intended status: Standards Track
Expires: January 8, 2017

X. Liu
Ericsson
Z. Zhang
ZTE Corporation
A. Peter
Juniper Networks
M. Sivakumar
Cisco Systems
F. Guo
Huawei Technologies
P. McAllister
Metaswitch Networks
July 7, 2016

MSDP YANG module
draft-zhang-pim-msdp-yang-01

Abstract

This document defines a YANG data model for MSDP protocol configuration and operation.

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 http://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 January 8, 2017.

Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
2. Design of the Data Model 2
3. MSDP configuration 5
4. Notifications 5
5. MSDP YANG module 5
6. Contributors 20
7. Normative References 20
Authors' Addresses 21

1. Introduction

[RFC3618] introduces protocol definition of MSDP. This document defines a YANG data model for MSDP. The content is in keeping with [RFC3618].

2. Design of the Data Model

The msdp peer and source-active content is important part of MSDP. The augment should be added below routing-protocol.

```
module: ietf-msdp
augment /rt:routing/rt:control-plane-protocols:
   +--rw msdp!
      +--rw global
      |  +--rw connect-source?   if:interface-ref
      |  +--rw default-peer! {global-default-peer}?
      |  |  +--rw peer-addr        -> ../../../peers/peer/address
      |  |  +--rw prefix-policy?   string {global-default-peer-policy}?
      |  +--rw originating-rp
      |  |  +--rw interface?   if:interface-ref
      |  +--rw sa-filter
      |  |  +--rw in?    string
      |  |  +--rw out?   string
      |  +--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) {peer-key-chain}?
            |        +--rw key?         string
            +--rw enable?           boolean {peer-admin-enable}?
            +--rw connect-source?   if:interface-ref
            +--rw description?      string {peer-description}?
            +--rw mesh-group?       string
            +--rw peer-as?          string {peer-as}?
            +--rw sa-filter
            |  +--rw in?    string
            |  +--rw out?   string
            +--rw timer
            |  +--rw connect-retry-interval?   uint16 {peer-timer-connect-
retry}?
            |  +--rw holdtime-interval?        uint16 {peer-timer-holdtime}?
            |  +--rw keepalive-interval?       uint16 {peer-timer-keepalive}?
            +--rw ttl-threshold?    uint8
augment /rt:routing-state/rt:control-plane-protocols:
   +--ro msdp!
      +--ro global
      |  +--ro connect-source?   if:interface-ref
      |  +--ro default-peer! {global-default-peer}?
      |  |  +--ro peer-addr        -> ../../../peers/peer/address
      |  |  +--ro prefix-policy?   string {global-default-peer-policy}?
      |  +--ro originating-rp
      |  |  +--ro interface?   if:interface-ref
      |  +--ro sa-filter
      |  |  +--ro in?    string
      |  |  +--ro out?   string
      |  +--ro ttl-threshold?    uint8
      +--ro peers
      |  +--ro peer* [address]
      |     +--ro address                 inet:ipv4-address
      |     +--ro authentication
      |     |  +--ro (authentication-type)?
      |     |     +--:(key-chain) {peer-key-chain}?
      |     |     |  +--ro key-chain?   key-chain:key-chain-ref
      |     |     +--:(password) {peer-key-chain}?
      |     |        +--ro key?         string
      |     +--ro enable?                 boolean {peer-admin-enable}?
      |     +--ro connect-source?         if:interface-ref
      |     +--ro description?            string {peer-description}?
      |     +--ro mesh-group?             string
      |     +--ro peer-as?                string {peer-as}?
      |     +--ro sa-filter
      |     |  +--ro in?    string
      |     |  +--ro out?   string
```

```
      |     +--ro timer
```

```
      |     |  +--ro connect-retry-interval?   uint16 {peer-timer-connect-
retry}?
      |     |  +--ro holdtime-interval?        uint16 {peer-timer-holdtime}?
      |     |  +--ro keepalive-interval?       uint16 {peer-timer-keepalive}?
      |     +--ro ttl-threshold?          uint8
      |     +--ro session-state?          enumeration
      |     +--ro elapsed-time?           uint32
      |     +--ro connect-retry-expire?   uint32
      |     +--ro hold-expire?            uint32
      |     +--ro is-default-peer?        boolean
      |     +--ro keepalive-expire?       uint32
      |     +--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 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 msdp-clear-peer
```

```
   |  +---w input
```

```
|     +---w peer-address?   inet:ipv4-address
+---x msdp-clear-sa-cache {rpc-clear-sa-cache}?
   +---w input
      +---w entry!
      |  +---w group          inet:ipv4-address
      |  +---w source-addr?   union
      +---w peer-address?   inet:ipv4-address
      +---w peer-as?        string
```

3. MSDP configuration

The msdp peers should be configured. And several peers may be in a mesh-group. The Source-Active information may be filtered for peers.

4. Notifications

This part will be updated in later version.

5. MSDP YANG module

```
<CODE BEGINS> file "ietf-msdp@2016-07-06.yang"
module ietf-msdp {
  namespace "urn:ietf:params:xml:ns:yang:ietf-msdp";
  // replace with IANA namespace when assigned
  prefix msdp;

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

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

  import ietf-routing {
    prefix "rt";
  }

  import ietf-interfaces {
    prefix "if";
  }

  import ietf-ip {
    prefix "ip";
  }

  import ietf-key-chain {
    prefix "key-chain";
```

```
     }

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

     contact
       "WG Web:   <http://tools.ietf.org/wg/pim/>
        WG List:  <mailto:pim@ietf.org>
        WG Chair: Stig Venaas
                  <mailto:stig@venaas.com>
        WG Chair: Mike McBride
                  <mailto:mmcbride7@gmail.com>

        Editors:   ";

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

     revision 2016-07-06 {
       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-connect-source {
       description
         "Support configuration of global connect-source.";
     }

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

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

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

```
     }

     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-connect-source {
       description
         "Support configuration of global 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 key-chain.";
     }

     feature peer-timer-connect-retry {
       description
         "Support configuration of peer timer for connect-retry.";
     }

     feature peer-timer-keepalive {
```

```
      description
        "Support configuration of peer timer for keepalive.";
    }

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

    /*
     * Typedefs
     */

    /*
     * Identities
     */

    /*
     * Groupings
     */
    grouping authentication-container {
      description
        "A container defining 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 {
            if-feature peer-key-chain;
            leaf key {
              type string;
              description
                "This leaf describes the authentication key.";
            }
            uses key-chain:crypto-algorithm-types;
          }
          description
            "Choice of authentication.";
        }
      }
```

```
     } // authentication-container

     grouping connect-source {
       description "Attribute to configure connect-source.";
       leaf connect-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.";
       }
     } // connect-source

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

       uses connect-source {
         if-feature global-connect-source;
       }
       container default-peer {
         if-feature global-default-peer;
         presence "";
         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 {
           if-feature global-default-peer-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;
        }
        uses ttl-threshold {
          if-feature global-ttl-threshold;
        }
      } // global-config-attributes

      grouping global-state-attributes {
        description "Global MSDP state attributes.";
      } // global-state-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 to enable peer;
```

```
            false to disable peer.";
        }
        uses connect-source {
          if-feature peer-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 string;
          description
            "Peer's autonomous system number (ASN).";
        }
        uses sa-filter-container;
        container timer {
          description "Timer attributes.";
          leaf connect-retry-interval {
            if-feature peer-timer-connect-retry;
            type uint16;
            units seconds;
            default 30;
            description "SHOULD be set to 30 seconds. ";
          }
          leaf holdtime-interval {
            if-feature peer-timer-holdtime;
            type uint16;
            units seconds;
            must ". > 3";
            default 75;
            description "The SA-Hold-Down-Period of this msdp peer.";
          }
          leaf keepalive-interval {
            if-feature peer-timer-keepalive;
            type uint16;
            units seconds;
            must ". > 1 and . < ../holdtime-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.";
          }
        }
        description
          "Peer session state.";
        reference
          "RFC3618: Multicast Source Discovery Protocol (MSDP).";
      }
      leaf elapsed-time {
        type uint32;
        units seconds;
        description "Elapsed time for being in a state.";
      }
      leaf connect-retry-expire {
        type uint32;
        units seconds;
        description "Connect retry expire time of peer connection.";
      }
      leaf hold-expire {
        type uint32;
        units seconds;
        description "Hold expire time of peer connection.";
      }
      leaf is-default-peer {
        type boolean;
        description "If this peer is default peer.";
```

```
        }
        leaf keepalive-expire {
          type uint32;
          units seconds;
          description "Keepalive expire time of this peer.";
        }
        leaf reset-count {
          type uint32;
          description "The reset count of this peer.";
        }
        uses statistics-container;
      } // peer-config-attributes

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

        leaf up-time {
          type uint32;
          units seconds;
          description "The up time of this sa cache.";
        }
        leaf expire {
          type uint32;
          units seconds;
          description "If this cache has expired.";
        }
        leaf holddown-interval {
          type uint32;
          units seconds;
          description "Holddown timer value for SA forwarding.";
        }
        leaf peer-learned-from {
          type inet:ipv4-address;
          description
            "The address of peer that we learned this SA from .";
        }
        leaf rpf-peer {
          type inet:ipv4-address;
          description "RPF peer.";
        }
      } // sa-cache-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.";
        }
        leaf out {
          type string;
          description
            "Filters outgoing SA messages only.";
        }
      } // 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-container {
      description
        "A container defining statistics attributes.";
      container statistics {
        description "";
        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 "";
          uses statistics-error;
        }
        container queue {
          description "";
          uses statistics-queue;
```

```
        }
        container received {
          description "";
          uses statistics-sent-received;
        }
        container sent {
          description "";
          uses statistics-sent-received;
        }
      }
    } // statistics-container

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

    grouping statistics-queue {
      description
        "A grouping defining 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

    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

      /*
       * Configuration data nodes
       */
      augment "/rt:routing/rt:control-plane-protocols" {
        description
          "MSDP augmentation to routing instance configuration.";

        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
                "";
            }
            uses peer-config-attributes;
          } // peer
        } // peers
      } // msdp
    } // augment

    /*
     * Operational state data nodes
     */
    augment "/rt:routing-state/rt:control-plane-protocols" {
      description
        "MSDP augmentation to routing instance state.";

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

        container global {
          description
            "Global attributes.";
          uses global-config-attributes;
          uses global-state-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 {
          description
            "The sa cache information.";
          list entry {
            key "group source-addr";
            description "";
            leaf group {
              type inet:ipv4-address;
              description "The group address of this sa cache.";
            }
            leaf source-addr {
              type union {
                type enumeration {
                  enum '*' {
                    description "The source addr of this sa cache.";
                  }
                }
                type inet:ipv4-address;
              }
              description "";
            }
            list origin-rp {
              key "rp-address";
              description
                "";
              leaf rp-address {
                type inet:ip-address;
                description "The rp address.";
              }
              leaf is-local-rp {
                type boolean;
                description "";
              }
              leaf sa-adv-expire {
                type uint32;
                units seconds;
                description
                  "Periodic SA advertisement timer exipiring time on
                   a local RP.";
              }
            }
            uses sa-cache-state-attributes;
          } // entry
        } // sa-cache
```

```
      } // msdp
    } // augment

    /*
     * RPCs
     */
    rpc msdp-clear-peer {
      description
        "Clears the session 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 msdp-clear-sa-cache {
      if-feature rpc-clear-sa-cache;
      description
        "Clears MSDP source active (SA) cache entries.";
      input {
        container entry {
          presence "";
          description
            "The SA cache (S,G) or (*,G) entry to be cleared. If this
             is not provided, all entries are cleared.";
          leaf group {
            type inet:ipv4-address;
            mandatory true;
            description "";
          }
          leaf source-addr {
            type union {
              type enumeration {
                enum '*' {
                  description "";
                }
              }
              type inet:ipv4-address;
            }
            description "";
          }
        } // 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 string;
          description
            "ASN from which MSDP SA cache entries have been learned.
             If this is not provided, entries learned from all AS's
             are cleared.";
        }
      }
    }

    /*
     * Notifications
     */
  }
  <CODE ENDS>
```

6. 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.

7. Normative References

[I-D.ietf-netmod-routing-cfg]
Lhotka, L. and A. Lindem, "A YANG Data Model for Routing Management", draft-ietf-netmod-routing-cfg-22 (work in progress), July 2016.

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

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

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

Authors' Addresses

Xufeng Liu
Ericsson
1595 Spring Hill Road, Suite 500
Vienna VA 22182
USA

Email: xliu@kuatrotech.com

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

Email: zhang.zheng@zte.com.cn

Anish Peter
Juniper Networks
Electra, Exora Business Park
Bangalore, KA 560103
India

Email: anishp@juniper.net

Mahesh Sivakumar
Cisco Systems
510 McCarthy Boulevard
Milpitas, California
USA

Email: masivaku@cisco.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