Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: October 19, 2015

Y. Liu
F. Guo
Huawei

April 17, 2015

Yang Data Model for Multicast in MPLS/BGP IP VPNs
draft-liu-bess-mvpn-yang-00

Abstract

This document defines a YANG data model that can be used to configure and manage MVPN.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

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 October 19, 2015.

Copyright Notice

Copyright (c) 2015 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. Terminology 3
3. Design of Data Model 3
3.1. Overview 3
3.2. MVPN Per-instance Configuration 5
3.2.1. Per-instance Parameters 5
3.2.2. Per-vpn-target Configuration of MVPN Instance 5
3.2.3. Per-tunnel-interface Configuration of MVPN Instance . 6
3.2.4. Per-share-mdt Configuration of MVPN Instance 6
3.2.5. Per-ipmsi-tunnel Configuration of MVPN Instance . . . 6
3.2.6. Per-switch-mdt Configuration of MVPN Instance 7
3.2.7. Per-spmsi-tunnel Configuration of MVPN Instance . . . 7
3.3. MVPN Operation State 8
4. MVPN Yang Module 10
5. IANA Considerations 34
6. Security Considerations 35
7. Acknowledgements 35
8. References 35
8.1. Normative References 35
8.2. Informative References 35
Authors' Addresses 35

1. Introduction

YANG[RFC6020] is a data definition language that was introduced to define the contents of a conceptual data store that allows networked devices to be managed using NETCONF[RFC6241]. YANG is proving relevant beyond its initial confines, as bindings to other interfaces(e.g. REST) and encoding other than XML (e.g. JSON) are being defined. Furthermore, YANG data models can be used as the basis of implementation for other interface, such as CLI and Programmatic APIs.

This document defines a YANG data model that can be used to configure and manage MVPN. It includes Cisco systems' solution [RFC6037], BGP MVPN [RFC6513][RFC6514] etc. In addition, features described in MVPN standards other than mentioned above RFC in future version.

2. Terminology

- o PIM: Protocol Independent Multicast
- o SM: Sparse Mode
- o SSM: Source-specific Multicast
- o PMSI: P-Multicast Service Interface
- o MDT: Multicast Distribution Tree
- o P2MP: Point to Multipoint
- o TE: Traffic Engineering
- o MLDP: Multipoint Label Distribution Protocol

3. Design of Data Model

3.1. Overview

MVPN includes two types of configuration objects and operation states for MD MVPN and NG MVPN. The MVPN Yang module is divided into five main containers, one contains the writable parameters, while the others contain the operational states.

- o mvpn-instances: that contains per-instance writable configuration objects.
- o mvpn-share-grp-infos: provide share group states information for MD MVPN instances.
- o mvpn-swt-grp-send-infos: provide switch group states sending information for MD MVPN instances.
- o mvpn-swt-grp-recv-infos: provide switch group states received information for MD MVPN instances.
- o mvpn-pmsi-infos: provide i-pmsi and s-pmsi states information for NG MVPN instances.

The figure below describes the overall structure of the MVPN Yang module :

```
module: mvpn
   +--rw mvpn-instances
   |  +--rw mvpn-instance* [vrf-name address-family]
```

```
   |     +--...
   |     +--rw mvpn-vpn-targets
   |     |  +--rw mvpn-vpn-target* [rt-type vpn-tar-val]
   |     |     +--...
   |     +--rw mvpn-tunnel-interfaces
   |     |  +--rw mvpn-tunnel-interface* [mtunnel-num]
   |     |     +--...
   |     +--rw mvpn-share-mdts
   |     |  +--rw mvpn-share-mdt
   |     |     +--...
   |     +--rw mvpn-ipmsi-tunnels
   |     |  +--rw mvpn-ipmsi-tunnel* [tunnel-mode]
   |     |     +--...
   |     +--rw mvpn-switch-mdts
   |     |  +--rw mvpn-switch-mdt
   |     |     +--...
   |     +--rw mvpn-spmsi-tunnels
   |        +--rw mvpn-spmsi-tunnel
   |           +--...
   +--ro mvpn-share-grp-infos
   |  +--ro mvpn-share-grp-info* [vrf-name address-family]
   |     +--...
   +--ro mvpn-swt-grp-send-infos
   |  +--ro mvpn-swt-grp-send-info* [vrf-name address-family]
   |     +--...
   |     +--ro mvpn-swt-sg-send-infos
   |        +--ro mvpn-swt-sg-send-info* [ipv4-src-address ipv4-grp-address]
   |           +--...
   +--ro mvpn-swt-grp-recv-infos
   |  +--ro mvpn-swt-grp-recv-info* [vrf-name address-family]
   |     +--...
   |     +--ro sender-infos
   |        +--ro sender-info* [ipv4-sender-address]
   |           +--...
   |           +--ro swt-sg-infos
   |              +--ro swt-sg-info* [ipv4-src-address ipv4-grp-address]
   |                 +--...
   +--ro mvpn-pmsi-infos
      +--ro mvpn-pmsi-info* [vrf-name address-family]
         +--...
         +--ro mvpn-ipmsi-tnl-infos
         |  +--ro mvpn-ipmsi-tnl-info* [tunnel-address]
         |     +--...
         +--ro mvpn-spmsi-sg-infos
            +--ro mvpn-spmsi-sg-info* [ipv4-src-address ipv4-grp-address]
               +--...
```

Figure 1 The overview of MVPN YANG data model

3.2. MVPN Per-instance Configuration

MVPN per-instance configuration container includes parameters of the MVPN instance binding a specific VRF. MVPN per-instance configuration container is divided into:

- o Per-instance parameters.
- o Per-vpn-target configuration of the MVPN instance.
- o Per-tunnel-interface configuration of the MVPN instance.
- o Per-share-mdt configuration of the MVPN instance.
- o Per-ipmsi-tunnel configuration of the MVPN instance.
- o Per-switch-mdt configuration of the MVPN instance.
- o Per-spmsi-tunnel configuration of the MVPN instance.

3.2.1. Per-instance Parameters

The per-instance parameter includes the name of the VRF bound by the MVPN instance, and signal mode or auto discovery mode etc.

```
+--rw mvpn-instances
|  +--rw mvpn-instance* [vrf-name address-family]
|     +--rw vrf-name                  string
|     +--rw address-family            enumeration
|     +--rw signaling-mode?           enumeration
|     +--rw auto-discovery-mode?      enumeration
|     +--rw config-type?              enumeration
|     +--rw is-sender-site?           boolean
|     +--rw rpt-spt-mode?             boolean
```

Figure 2 The YANG data model of MVPN instance configuration

3.2.2. Per-vpn-target Configuration of MVPN Instance

The vpn-target of mvpn can be configured differently from unicast vpn. By default, unicast vpn-targets are always used for multicast vpn.

```
|     +--rw mvpn-vpn-targets
|     |  +--rw mvpn-vpn-target* [rt-type vpn-tar-val]
|     |     +--rw rt-type        enumeration
|     |     +--rw vpn-tar-val    string
```

Figure 3 The YANG data model of MVPN vpn-target configuration

3.2.3. Per-tunnel-interface Configuration of MVPN Instance

The tunnel interface configuration includes interface number or name, and address for ipv4 or ipv6 etc. It can borrow address from other interfaces.

```
|     +--rw mvpn-tunnel-interfaces
|     |  +--rw mvpn-tunnel-interface* [mtunnel-num]
|     |     +--rw mtunnel-num                  uint16
|     |     +--rw mtunnel-name?                string
|     |     +--rw is-unnumbered?               boolean
|     |     +--rw unnumbered-if-name?          if-name
|     |     +--rw mtunnel-if-ipv4-address?     inet:ipv4-address
|     |     +--rw mtunnel-if-ipv6-address?     inet:ipv6-address
|     |     +--rw mtunnel-if-ipv4-masklength?  uint8
|     |     +--rw mtunnel-if-ipv6-masklength?  uint8
```

Figure 4 The YANG data model of MVPN tunnel-interface configuration

3.2.4. Per-share-mdt Configuration of MVPN Instance

For MD MVPN, every vpn instance has only one share mdt, and it can be configured pim mode and group address of the multicast tree in public net.

```
|     +--rw mvpn-share-mdts
|     |  +--rw mvpn-share-mdt
|     |     +--rw share-mdt-mode         enumeration
|     |     +--rw ipv4-share-grp-addr    inet:ipv4-address
|     |     +--rw ipv6-share-grp-addr    inet:ipv6-address
```

Figure 5 The YANG data model of MVPN share-mdt configuration

3.2.5. Per-ipmsi-tunnel Configuration of MVPN Instance

For NG MVPN, every vpn instance has only one i-pmsi tunnel, and it can be configured tunnel mode, and especially specified template for p2mp te.

```
|     +--rw mvpn-ipmsi-tunnels
|     |  +--rw mvpn-ipmsi-tunnel* [tunnel-mode]
|     |     +--rw tunnel-mode         enumeration
|     |     +--rw te-p2mp-template?   string
```

Figure 6 The YANG data model of MVPN ipmsi-tunnel configuration

3.2.6. Per-switch-mdt Configuration of MVPN Instance

For MD MVPN, every vpn instance can be configured switch group address pool for forwarding multicast data from share mdt to switch mdt. It can be also specified switch threshold value, group policies etc.

```
|     +--rw mvpn-switch-mdts
|     |  +--rw mvpn-switch-mdt
|     |     +--rw mdt-limit?                    uint16
|     |     +--rw switch-mdt-mode               enumeration
|     |     +--rw ipv4-group-pool-addr          inet:ipv4-address
|     |     +--rw ipv4-group-pool-masklength    uint8
|     |     +--rw ipv6-group-pool-addr          inet:ipv6-address
|     |     +--rw ipv6-group-pool-masklength    uint8
|     |     +--rw switch-threshold?             uint32
|     |     +--rw switch-delay-time?            uint8
|     |     +--rw hold-down-time?               uint8
|     |     +--rw group-acl-name?               string
|     |     +--rw group-acl-ipv6-name?          string
|     |     +--rw is-switch-group-reuse-log?    boolean
```

Figure 7 The YANG data model of MVPN switch-mdt configuration

3.2.7. Per-spmsi-tunnel Configuration of MVPN Instance

For NG MVPN, every vpn instance can be configured s-pmsi tunnels for forwarding multicast data from i-pmsi tunnel to s-pmsi tunnel. It can be also specified group and source address in private net, and switch threshold value etc.

```
|     +--rw mvpn-spmsi-tunnels
|        +--rw mvpn-spmsi-tunnel
|           +--rw ipv4-group-addr           inet:ipv4-address
|           +--rw ipv6-group-addr           inet:ipv6-address
|           +--rw ipv4-group-masklength     uint8
|           +--rw ipv6-groupmasklength      uint8
|           +--rw ipv4-source-addr          inet:ipv4-address
|           +--rw ipv6-source-addr          inet:ipv6-address
|           +--rw ipv4-source-masklength    uint8
|           +--rw ipv6-source-masklength    uint8
|           +--rw tunnel-mode               enumeration
|           +--rw p2mp-te-template?         string
|           +--rw threshold?                uint32
|           +--rw tunnel-limit?             uint8
```

Figure 8 The YANG data model of MVPN spmsi-tunnel configuration

3.3. MVPN Operation State

MVPN container provides operational states for MD MVPN and NG MVPN instances. This container is divided in four components:

- o mvpn-share-grp-infos: provide share group states information for MD MVPN instances.

- o mvpn-swt-grp-send-infos: provide switch group states sending information for MD MVPN instances.

- o mvpn-swt-grp-recv-infos: provide switch group states received information for MD MVPN instances.

- o mvpn-pmsi-infos: provide i-pmsi and s-pmsi states information for NG MVPN instances.

```
+--ro mvpn-share-grp-infos
|  +--ro mvpn-share-grp-info* [vrf-name address-family]
|     +--ro vrf-name                  string
|     +--ro address-family            enumeration
|     +--ro ipv4-share-grp-address?   inet:ipv4-address
|     +--ro ipv6-share-grp-address?   inet:ipv6-address
|     +--ro ipv4-mtunnel-address?     inet:ipv4-address
|     +--ro ipv6-mtunnel-address?     inet:ipv6-address
```

Figure 9 The YANG data model of MVPN share group information

```
+--ro mvpn-swt-grp-send-infos
|  +--ro mvpn-swt-grp-send-info* [vrf-name address-family]
|     +--ro vrf-name                  string
|     +--ro address-family            enumeration
|     +--ro ipv4-swt-grp-address?     inet:ipv4-address
|     +--ro ipv6-swt-grp-address?     inet:ipv6-address
|     +--ro reference-count?          uint32
|     +--ro mvpn-swt-sg-send-infos
|        +--ro mvpn-swt-sg-send-info* [ipv4-src-address ipv4-grp-address]
|           +--ro ipv4-src-address    inet:ipv4-address
|           +--ro ipv6-src-address?   inet:ipv6-address
|           +--ro ipv4-grp-address    inet:ipv4-address
|           +--ro ipv6-grp-address?   inet:ipv6-address
|           +--ro swt-time?           uint32
```

Figure 10 The YANG data model of MVPN switch group sending information

```
+--ro mvpn-swt-grp-recv-infos
|  +--ro mvpn-swt-grp-recv-info* [vrf-name address-family]
|     +--ro vrf-name                string
|     +--ro address-family          enumeration
|     +--ro ipv4-swt-grp-address?   inet:ipv4-address
|     +--ro ipv6-swt-grp-address?   inet:ipv6-address
|     +--ro reference-count?        uint32
|     +--ro active-count?           uint32
|     +--ro sender-infos
|        +--ro sender-info* [ipv4-sender-address]
|           +--ro ipv4-sender-address    inet:ipv4-address
|           +--ro ipv6-sender-address?   inet:ipv6-address
|           +--ro sender-active-cnt?     uint32
|           +--ro swt-sg-infos
|              +--ro swt-sg-info* [ipv4-src-address ipv4-grp-address]
|                 +--ro ipv4-src-address    inet:ipv4-address
|                 +--ro ipv6-src-address?   inet:ipv6-address
|                 +--ro ipv4-grp-address    inet:ipv4-address
|                 +--ro ipv6-grp-address?   inet:ipv6-address
|                 +--ro expire-time?        uint32
|                 +--ro is-active?          boolean
```

Figure 11 The YANG data model of MVPN switch group received information

```
   +--ro mvpn-pmsi-infos
      +--ro mvpn-pmsi-info* [vrf-name address-family]
         +--ro vrf-name                string
         +--ro address-family          enumeration
         +--ro mvpn-pmsi-type          enumeration
         +--ro tunnel-mode             enumeration
         +--ro te-p2mp-id?             uint16
         +--ro te-tunnel-id?           uint16
         +--ro te-extend-tunnel-id?    uint16
         +--ro mldp-root-addr?         inet:ipv4-address
         +--ro mldp-lsp-id?            string
         +--ro mvpn-ipmsi-tnl-infos
         |  +--ro mvpn-ipmsi-tnl-info* [tunnel-address]
         |     +--ro tunnel-address    inet:ipv4-address
         |     +--ro tunnel-role?      enumeration
         +--ro mvpn-spmsi-sg-infos
            +--ro mvpn-spmsi-sg-info* [ipv4-src-address ipv4-grp-address]
               +--ro ipv4-src-address    inet:ipv4-address
               +--ro ipv6-src-address?   inet:ipv6-address
               +--ro ipv4-grp-address    inet:ipv4-address
               +--ro ipv6-grp-address?   inet:ipv6-address
```

Figure 12 The YANG data model of MVPN i-pmsi and s-pmsi information

4. MVPN Yang Module

```
module mvpn {
    namespace "urn:huawei:params:xml:ns:yang:mvpn";
    // replace with IANA namespace when assigned -
    // urn:ietf:params:xml:ns:yang:1
    prefix "mvpn";
    import ietf-inet-types {
        prefix inet;
    }
    organization
      "IETF BESS(BGP Enabled Services) Working Group";
    contact
      "liuyisong@huawei.com
       guofeng@huawei.com";
    description
      "This YANG module defines the generic configuration
       data for mvpn, which is common across all of the vendor
       implementations of the protocol. It is intended that the module
       will be extended by vendors to define vendor-specific
       mvpn configuration parameters.";
    revision 2015-04-30 {
        description
          "Initial revision.";
```

```
    }

    typedef if-name {
        description "if-name is like ethernet1/1/1/1";
        type string {
            length "1..63";
        }
    }




    container mvpn-instances {

        list mvpn-instance {

            key "vrf-name address-family";
            config "true";

            leaf vrf-name {
                description "Name of a vpn instance. ";
                config "true";
                type string {
                    length "0..32";
                }
            }
            leaf address-family {
                config "true";
                mandatory "true";
                type enumeration {
                    enum ipv4unicast {
                        value "0";
                        description "ipv4unicast:";
                    }
                    enum ipv6unicast {
                        value "1";
                        description "ipv6unicast:";
                    }
                }
            }
            leaf signaling-mode {
                description "Signaling mode.";
                config "true";
                default "invalid";
                type enumeration {
                    enum invalid {
                        value "0";
                        description "invalid:invalid";
```

```
                }
                enum bgp {
                    value "1";
                    description "bgp:bgp";
                }
                enum pim {
                    value "2";
                    description "pim:pim";
                }
            }
        }
        leaf auto-discovery-mode {
            description "Auto discovery mode.";
            config "true";
            default "none";
            type enumeration {
                enum none {
                    value "0";
                    description "none:none";
                }
                enum ad {
                    value "1";
                    description "ad:ad";
                }
            }
        }
        leaf config-type {
            description "Mvpn type, which can be md or ng.";
            config "true";
            default "md";
            type enumeration {
                enum md {
                    value "0";
                    description "md:md";
                }
                enum ng {
                    value "1";
                    description "ng:ng";
                }
            }
        }
        leaf is-sender-site {
            description "Configure the current PE as a sender PE.";
            config "true";
            default "false";
            type boolean{
            }
        }
```

```
            leaf rpt-spt-mode {
                description "Rpt and spt mode in private net.";
                config "true";
                default "false";
                type boolean{
                }
            }

            container mvpn-vpn-targets {

                list mvpn-vpn-target {

                    key "rt-type vpn-tar-val";

                    leaf rt-type {
                        description
                            "rt types are as follows:
                             export-extcommunity: specifies the value of
                             the extended community attribute of the
                             route from an outbound interface to the
                             destination vpn.
                             import-extcommunity: receives routes that
                             carry the specified extended community
                             attribute";
                        config "true";
                        mandatory "true";
                        type enumeration {
                            enum export-extcommunity {
                                value "0";
                                description "export-extcommunity";
                            }
                            enum import-extcommunity {
                                value "1";
                                description "import-extcommunity";
                            }
                        }
                    }
                    leaf vpn-tar-val {
                        description
                            "the available mvpn target formats are as
                             follows:
                             - 16-bit as number:32-bit user-defined
                             number, for example, 1:3. an as number
                             ranges from 0 to 65535, and a user-defined
                             number ranges from 0 to 4294967295. The as
                             number and user-defined number cannot be
                             both 0s. That is, a vpn target cannot be 0:0.
                             - 32-bit ip address:16-bit user-defined
```

```
                         number, for example, 192.168.122.15:1.
                         The ip address ranges from 0.0.0.0 to
                         255.255.255.255, and the user-defined
                         number ranges from 0 to 65535.";
                     config "true";
                     type string {
                         length "3..21";
                     }
                 }
             }

         }

         container mvpn-tunnel-interfaces {

             list mvpn-tunnel-interface {

                 key "mtunnel-num";

                 leaf mtunnel-num {
                     description "Mti id.";
                     config "true";
                     type uint16 {
                         range "0..4095";
                     }
                 }
                 leaf mtunnel-name {
                     description "Mti name.";
                     config "true";
                     type string {
                         length "1..48";
                     }
                 }
                 leaf is-unnumbered {
                     description
                         "Configure the ip address of a specified
                          interface as the default ip address of
                          an mti.";
                     config "true";
                     default "false";
                     type boolean{
                     }
                 }
                 leaf unnumbered-if-name {
                     description
                         "Interface whose ip address is used as the
                          default ip address of an mti.";
                     config "true";
```

```
                    type if-name;
                }
                leaf mtunnel-if-ipv4-address {
                    description "Mti ipv4 address.";
                    config "true";
                    type inet:ipv4-address;
                }
                leaf mtunnel-if-ipv6-address {
                    description "Mti ipv6 address.";
                    config "true";
                    type inet:ipv6-address;
                }
                leaf mtunnel-if-ipv4-masklength {
                    description "Mask length of the mti ipv4 address.";
                    config "true";
                    type uint8 {
                        range "0..32";
                    }
                }
                leaf mtunnel-if-ipv6-masklength {
                    description "Mask length of the mti ipv6 address.";
                    config "true";
                    type uint8 {
                        range "0..128";
                    }
                }
            }

        }

        container mvpn-share-mdts {

            container mvpn-share-mdt {

                 leaf share-mdt-mode {
                    description "Share mdt mode.";
                    config "true";
                    mandatory "true";
                    type enumeration {
                        enum invalid {
                            value "0";
                            description "invalid:invalid";
                        }
                        enum pim-ssm {
                            value "1";
                            description "pim-ssm:pim-ssm";
                        }
                        enum pim-sm {
```

```
                                value "2";
                                description "pim-sm:pim-sm";
                            }
                            enum pim-dm {
                                value "3";
                                description "pim-dm:pim-dm";
                            }
                            enum bidir-pim {
                                value "4";
                                description "bidir-pim:bidir-pim";
                            }
                        }
                    }
                    leaf ipv4-share-grp-addr {
                        description "share-group ipv4 address.";
                        config "true";
                        mandatory "true";
                        type inet:ipv4-address;
                    }
                    leaf ipv6-share-grp-addr {
                        description "share-group ipv6 address.";
                        config "true";
                        mandatory "true";
                        type inet:ipv6-address;
                    }

                }

            }

            container mvpn-ipmsi-tunnels {

                list mvpn-ipmsi-tunnel {

                    key "tunnel-mode";

                    leaf tunnel-mode {
                        description "I-PMSI tunnel mode.";
                        config "true";
                        type enumeration {
                            enum invalid {
                                value "0";
                                description "invalid:invalid";
                            }
                            enum p2mp-te {
                                value "1";
                                description "p2mp-te:p2mp-te";
                            }
```

```
                        enum p2mp-mldp {
                            value "2";
                            description "p2mp-mldp:p2mp-mldp";
                        }
                        enum pim-ssm {
                            value "3";
                            description "pim-ssm:pim-ssm";
                        }
                        enum pim-sm {
                            value "4";
                            description "pim-sm:pim-sm";
                        }
                        enum bidir-pim {
                            value "5";
                            description "bidir-pim:bidir-pim";
                        }
                        enum ingress-replication {
                            value "6";
                            description "ingress-replication:
                                         ingress-replication";
                        }
                        enum mp2mp-mldp {
                            value "7";
                            description "mp2mp-mldp:mp2mp-mldp";
                        }
                    }
                }
                leaf te-p2mp-template {
                    description "P2mp te tunnel template";
                    config "true";
                    type string {
                        length "1..31";
                    }
                }
            }

        }

        container mvpn-switch-mdts {

            container mvpn-switch-mdt {

                 leaf mdt-limit {
                    description
                        "maximum number of switch mdts allowed.";
                    config "true";
                    default "0";
                    type uint16 {
```

```
                        range "0..256";
                    }
                }
                leaf switch-mdt-mode {
                    description "Switch mdt mode.";
                    config "true";
                    mandatory "true";
                    type enumeration {
                        enum invalid {
                            value "0";
                            description "invalid:invalid";
                        }
                        enum pim-ssm {
                            value "1";
                            description "pim-ssm:pim-ssm";
                        }
                        enum pim-sm {
                            value "2";
                            description "pim-sm:pim-sm";
                        }
                        enum pim-dm {
                            value "3";
                            description "pim-dm:pim-dm";
                        }
                        enum bidir-pim {
                            value "4";
                            description "bidir-pim:bidir-pim";
                        }
                    }
                }
                leaf ipv4-group-pool-addr {
                    description
                        "Start and end ipv4 addresses of the
                         switch-group address pool. The value is
                         in dotted decimal notation and ranges
                         from 224.0.1.0 to 239.255.255.255.";
                    config "true";
                    mandatory "true";
                    type inet:ipv4-address;
                }
                leaf ipv4-group-pool-masklength {
                    description
                        "Mask length for ipv4 addresses in the
                         switch-group address pool.";
                    config "true";
                    mandatory "true";
                    type uint8 {
                        range "24..32";
```

```
                    }
                }
                leaf ipv6-group-pool-addr {
                    description
                        "Start and end ipv6 addresses of the
                         switch-group address pool. ";
                    config "true";
                    mandatory "true";
                    type inet:ipv6-address;
                }
                leaf ipv6-group-pool-masklength {
                    description
                        "Mask length for ipv6 addresses in the
                         switch-group address pool.";
                    config "true";
                    mandatory "true";
                    type uint8 {
                        range "96..128";
                    }
                }
                leaf switch-threshold {
                    description
                        "Multicast packet rate threshold for
                         triggering the switching from the
                         share-mdt to the switch-mdt. The value is
                         an integer ranging from 0 to 4194304, in
                         kbit/s. The default value is 0.";
                    config "true";
                    default "0";
                    type uint32 {
                        range "0..4194304";
                    }
                }
                leaf switch-delay-time {
                    description
                        "Delay for switching from the share-mdt to
                         the switch-mdt. The value is an integer
                         ranging from 3 to 60, in seconds. ";
                    config "true";
                    default "5";
                    type uint8 {
                        range "3..60";
                    }
                }
                leaf hold-down-time {
                    description
                        "Delay for switching from the switch-mdt
                         to the share-mdt. The value is an integer
```

```
                         ranging from 0 to 512, in seconds. ";
                    config "true";
                    default "60";
                    type uint8 {
                        range "0..180";
                    }
                }
                leaf group-acl-name {
                    description
                        "Specify the (s, g) entry on which the
                         switch-group address pool takes effect.
                         The value is an integer ranging from 3000
                         to 3999 or a string of 32 case-sensitive
                         characters. If no value is specified, the
                         switch-group address pool takes effect on
                         all (s, g).";
                    config "true";
                    type string {
                        length "1..32";
                    }
                }
                leaf group-acl-ipv6-name {
                    description
                        "Specify the (s, g) entry on which the
                         switch-group address pool takes effect.
                         The value is an integer ranging from 3000
                         to 3999 or a string of 32 case-sensitive
                         characters. If no value is specified, the
                         switch-group address pool takes effect on
                         all (s, g).";
                    config "true";
                    type string {
                        length "1..32";
                    }
                }
                leaf is-switch-group-reuse-log {
                    description
                        "Enable the function to record switch-group
                         address reuse information.";
                    config "true";
                    default "false";
                    type boolean{
                    }
                }

            }

        }
```

```
            container mvpn-spmsi-tunnels {

                container mvpn-spmsi-tunnel {

                    leaf ipv4-group-addr {
                        description
                            "Start and end ipv4 addresses of the group
                             address in private net. The value is in
                             dotted decimal notation and ranges from
                             224.0.1.0 to 239.255.255.255.";
                        config "true";
                        mandatory "true";
                        type inet:ipv4-address;
                    }
                    leaf ipv6-group-addr {
                        description
                            "Start and end ipv6 addresses of the group
                             address in private net.";
                        config "true";
                        mandatory "true";
                        type inet:ipv6-address;
                    }
                    leaf ipv4-group-masklength {
                        description
                            "Group mask length for ipv4 addresses in
                             the group address pool in private net.";
                        config "true";
                        mandatory "true";
                        type uint8 {
                            range "4..32";
                        }
                    }
                    leaf ipv6-groupmasklength {
                        description
                            "Group mask length for ipv6 addresses in
                             the group address pool in private net.";
                        config "true";
                        mandatory "true";
                        type uint8 {
                            range "8..128";
                        }
                    }
                    leaf ipv4-source-addr {
                        description
                            "Start and end ipv4 addresses of the source
                             address in private net.";
                        config "true";
                        mandatory "true";
```

```
                        type inet:ipv4-address;
                    }
                    leaf ipv6-source-addr {
                        description
                            "Start and end ipv6 addresses of the source
                             address in private net.";
                        config "true";
                        mandatory "true";
                        type inet:ipv6-address;
                    }
                    leaf ipv4-source-masklength {
                        description
                            "Source mask length for ipv4 addresses in
                             the group address pool in private net.";
                        config "true";
                        mandatory "true";
                        type uint8 {
                            range "0..32";
                        }
                    }
                    leaf ipv6-source-masklength {
                        description
                            "Source mask length for ipv6 addresses in
                            the group address pool in private net.";
                        config "true";
                        mandatory "true";
                        type uint8 {
                            range "0..128";
                        }
                    }
                    leaf tunnel-mode {
                        description "S-pmsi tunnel mode.";
                        config "true";
                        mandatory "true";
                        type enumeration {
                            enum invalid {
                                value "0";
                                description "invalid:invalid";
                            }
                            enum p2mp-te {
                                value "1";
                                description "p2mp-te:p2mp-te";
                            }
                            enum p2mp-mldp {
                                value "2";
                                description "p2mp-mldp:p2mp-mldp";
                            }
                            enum pim-ssm {
```

```
                    value "3";
                    description "pim-ssm:pim-ssm";
                }
                enum pim-sm {
                    value "4";
                    description "pim-sm:pim-sm";
                }
                enum bidir-pim {
                    value "5";
                    description "bidir-pim:bidir-pim";
                }
                enum ingress-replication {
                    value "6";
                    description "ingress-replication:
                                 ingress-replication";
                }
                enum mp2mp-mldp {
                    value "7";
                    description "mp2mp-mldp:mp2mp-mldp";
                }
            }
        }
        leaf p2mp-te-template {
            description "P2mp te tunnel template";
            config "true";
            type string {
                length "1..31";
            }
        }

        leaf threshold {
            description
                "Multicast packet rate threshold for
                 triggering the switching from the i-pmsi
                 to the s-pmsi. The value is an integer
                 ranging from 0 to 4194304, in kbit/s. The
                 default value is 0.";
            config "true";
            type uint32 {
                range "0..4194304";
            }
        }
        leaf tunnel-limit {
            description
                "Maximum number of s-pmsi tunnels allowed.";
            config "true";
            type uint8 {
                range "1..16";
```

```
                        }
                    }

                }
            }

        }

    }



    container mvpn-share-grp-infos {

        config "false";

        list mvpn-share-grp-info {

            key "vrf-name address-family";
            config "false";

            leaf vrf-name {
                description "Name of a vpn instance. ";
                config "false";
                type string {
                    length "0..32";
                }
            }
            leaf address-family {
                description "Ipv4 or ipv6 address family.";
                config "false";
                type enumeration {
                    enum ipv4unicast {
                        value "0";
                        description "ipv4unicast:";
                    }
                    enum ipv6unicast {
                        value "1";
                        description "ipv6unicast:";
                    }
                }
            }
            leaf ipv4-share-grp-address {
                description "Share-group address for ipv4.";
                config "false";
                type inet:ipv4-address;
            }
            leaf ipv6-share-grp-address {
```

```
                description "Share-group address for ipv6.";
                config "false";
                type inet:ipv6-address;
            }
            leaf ipv4-mtunnel-address {
                description "Mti ip address for ipv4.";
                config "false";
                type inet:ipv4-address;
            }
            leaf ipv6-mtunnel-address {
                description "Mti ip address for ipv6.";
                config "false";
                type inet:ipv6-address;
            }

        }

    }

    container mvpn-swt-grp-send-infos {

        config "false";

        list mvpn-swt-grp-send-info {

            key "vrf-name address-family";
            config "false";
            description
                "Switch-group information sent by a specified vpn
                 instance.";

            leaf vrf-name {
                description "Name of a vpn instance. ";
                config "false";
                type string {
                    length "0..32";
                }
            }
            leaf address-family {
                description "Ipv4 or ipv6 address family.";
                config "false";
                type enumeration {
                    enum ipv4unicast {
                        value "0";
                        description "ipv4unicast:";
                    }
                    enum ipv6unicast {
                        value "1";
```

```
                    description "ipv6unicast:";
                }
            }
        }
        leaf ipv4-swt-grp-address {
            description "Switch-group address for ipv4.";
            config "false";
            type inet:ipv4-address;
        }
        leaf ipv6-swt-grp-address {
            description "Switch-group address for ipv6.";
            config "false";
            type inet:ipv6-address;
        }
        leaf reference-count {
            description
                "Number of vpn multicast groups joining a
                 switch-group.";
            config "false";
            type uint32 {
                range "1..4294967295";
            }
        }
        container mvpn-swt-sg-send-infos {

            list mvpn-swt-sg-send-info {

                key "ipv4-src-address ipv4-grp-address";
                config "false";

                leaf ipv4-src-address {
                    description
                        "Source address on the mvpn for ipv4.";
                    config "false";
                    type inet:ipv4-address;
                }
                leaf ipv6-src-address {
                    description
                        "Source address on the mvpn for ipv6.";
                    config "false";
                    type inet:ipv6-address;
                }

                leaf ipv4-grp-address {
                    description
                        "Group address on the mvpn for ipv4.";
                    config "false";
                    type inet:ipv4-address;
```

```
                    }
                    leaf ipv6-grp-address {
                        description
                            "Group address on the mvpn for ipv6.";
                        config "false";
                        type inet:ipv6-address;
                    }

                    leaf swt-time {
                        description
                            "Switching duration of the vpn multicast
                             (s, g) entry joining the switch-group. ";
                        config "false";
                        type uint32 {
                            range "1..4294967295";
                        }
                    }
                }

            }

        }

    }

    container mvpn-swt-grp-recv-infos {

        config "false";

        list mvpn-swt-grp-recv-info {

            key "vrf-name address-family";
            config "false";
            description
                "Switch-group information received by a specified vpn
                 instance.";

            leaf vrf-name {
                description "Name of a vpn instance. ";
                config "false";
                type string {
                    length "0..32";
                }
            }
            leaf address-family {
                description "Ipv4 or ipv6 address family.";
                config "false";
                type enumeration {
```

```
                    enum ipv4unicast {
                        value "0";
                        description "ipv4unicast:";
                    }
                    enum ipv6unicast {
                        value "1";
                        description "ipv6unicast:";
                    }
                }
            }
            leaf ipv4-swt-grp-address {
                description "Switch-group address for ipv4.";
                config "false";
                type inet:ipv4-address;
            }
            leaf ipv6-swt-grp-address {
                description "Switch-group address for ipv6.";
                config "false";
                type inet:ipv6-address;
            }
            leaf reference-count {
                description
                    "Number of vpn multicast groups joining the
                     switch-group.";
                config "false";
                type uint32 {
                    range "1..4294967295";
                }
            }
            leaf active-count {
                description
                    "Number of active vpn multicast groups
                     (with receivers) joining a switch-group.";
                config "false";
                type uint32 {
                    range "1..4294967295";
                }
            }
            container sender-infos {

                list sender-info {

                    key "ipv4-sender-address";
                    config "false";

                    leaf ipv4-sender-address {
                        description
                            "Bgp peer address of the PE that sends
```

```
                 switch-group information for ipv4.";
            config "false";
            type inet:ipv4-address;
        }
        leaf ipv6-sender-address {
            description
                "Bgp peer address of the PE that sends
                 switch-group information for ipv6.";
            config "false";
            type inet:ipv6-address;
        }
        leaf sender-active-cnt {
            description
                "Number of active vpn multicast groups
                 (with receivers) referenced by a sender.";
            config "false";
            type uint32 {
                range "1..4294967295";
            }
        }
        container swt-sg-infos {

            list swt-sg-info {

                key "ipv4-src-address ipv4-grp-address";
                config "false";

                leaf ipv4-src-address {
                    description
                        "Mvpn source address for ipv4.";
                    config "false";
                    type inet:ipv4-address;
                }
                leaf ipv6-src-address {
                    description
                        "Mvpn source address for ipv6.";
                    config "false";
                    type inet:ipv6-address;
                }
                leaf ipv4-grp-address {
                    description
                        "Mvpn group address for ipv4.";
                    config "false";
                    type inet:ipv4-address;
                }
                leaf ipv6-grp-address {
                    description
                        "Mvpn group address for ipv6.";
```

```
                                config "false";
                                type inet:ipv6-address;
                            }
                            leaf expire-time {
                                description
                                    "Timeout duration of vpn (s, g)
                                     entries.";
                                config "false";
                                type uint32 {
                                    range "1..4294967295";
                                }
                            }
                            leaf is-active {
                                description
                                    "Flag indicating whether a vpn
                                     (s, g) entry is active.";
                                config "false";
                                type boolean{
                                }
                            }
                        }

                    }

                }

            }

        }

    }

    container mvpn-pmsi-infos {

        config "false";

        list mvpn-pmsi-info {

            key "vrf-name address-family";
            config "false";
            description "i-pmsi & s-pmsi information.";

            leaf vrf-name {
                description "Name of a vpn instance. ";
                config "false";
                type string {
                    length "0..32";
                }
```

```
            }
            leaf address-family {
                description "ipv4 or ipv6 address family.";
                config "false";
                type enumeration {
                    enum ipv4unicast {
                        value "0";
                        description "ipv4unicast:";
                    }
                    enum ipv6unicast {
                        value "1";
                        description "ipv6unicast:";
                    }
                }
            }
            leaf mvpn-pmsi-type {
                description "Type of mvpn pmsi";
                config "false";
                mandatory "true";
                type enumeration {
                    enum i-pmsi{
                        value "0";
                        description "i-pmsi information";
                    }
                    enum s-pmsi{
                        value "1";
                        description "s-pmsi information";
                    }
                }
            }
            leaf tunnel-mode {
                description "Tunnel mode.";
                config "false";
                mandatory "true";
                type enumeration {
                    enum invalid {
                        value "0";
                        description "invalid:invalid";
                    }
                    enum p2mp-te {
                        value "1";
                        description "p2mp-te:p2mp-te";
                    }
                    enum p2mp-mldp {
                        value "2";
                        description "p2mp-mldp:p2mp-mldp";
                    }
                    enum pim-ssm {
```

```
                        value "3";
                        description "pim-ssm:pim-ssm";
                    }
                    enum pim-sm {
                        value "4";
                        description "pim-sm:pim-sm";
                    }
                    enum bidir-pim {
                        value "5";
                        description "bidir-pim:bidir-pim";
                    }
                    enum ingress-replication {
                        value "6";
                        description "ingress-replication:
                                     ingress-replication";
                    }
                    enum mp2mp-mldp {
                        value "7";
                        description "mp2mp-mldp:mp2mp-mldp";
                    }
                }
            }
            leaf te-p2mp-id {
                description "P2mp id of the p2mp tunnel.";
                config "false";
                default "0";
                type uint16 {
                    range "0..65535";
                }
            }
            leaf te-tunnel-id {
                description "Id of the p2mp tunnel.";
                config "false";
                type uint16 {
                    range "1..65535";
                }
            }
            leaf te-extend-tunnel-id {
                description "P2mp extended tunnel interface id.";
                config "false";
                type uint16 {
                    range "1..65535";
                }
            }
            leaf mldp-root-addr {
                description "Ip address of the root of a p2mp ldp lsp.";
                config "false";
                type inet:ipv4-address;
```

```
            }
            leaf mldp-lsp-id {
                description "P2mp ldp lsp id.";
                config "false";
                type string {
                    length "1..256";
                }
            }
            container mvpn-ipmsi-tnl-infos {

                list mvpn-ipmsi-tnl-info {

                    key "tunnel-address";
                    config "false";

                    leaf tunnel-address {
                        description "Tunnel node address.";
                        config "false";
                        type inet:ipv4-address;
                    }
                    leaf tunnel-role {
                        description "Role of a tunnel node.";
                        config "false";
                        type enumeration {
                            enum none {
                                value "0";
                                description "none:nome";
                            }
                            enum root {
                                value "1";
                                description "root:root";
                            }
                            enum leaf {
                                value "2";
                                description "leaf:leaf";
                            }
                            enum root-and-leaf {
                                value "3";
                                description "root-and-leaf:
                                             root-and-leaf";
                            }
                        }
                    }
                }

            }
            container mvpn-spmsi-sg-infos {
```

```
                list mvpn-spmsi-sg-info {

                    key "ipv4-src-address ipv4-grp-address";
                    config "false";

                    leaf ipv4-src-address {
                        description
                            "Source address in s-pmsi for ipv4.";
                        config "false";
                        type inet:ipv4-address;
                    }
                    leaf ipv6-src-address {
                        description
                            "Source address in s-pmsi for ipv6.";
                        config "false";
                        type inet:ipv6-address;
                    }
                    leaf ipv4-grp-address {
                        description
                            "Group address in s-pmsi for ipv4.";
                        config "false";
                        type inet:ipv4-address;
                    }
                    leaf ipv6-grp-address {
                        description
                            "Group address in s-pmsi for ipv6.";
                        config "false";
                        type inet:ipv6-address;
                    }

                }

            }

        }

    }

}
```

5. IANA Considerations

This draft includes no request to IANA.

6. Security Considerations

The data model defined does not create any security implications. This draft does not change any underlying security issues inherent in [I-D.ietf-netmod-routing-cfg].

7. Acknowledgements

TBD

8. References

8.1. 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-17 (work in progress), March 2015.

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[RFC6037] Rosen, E., Cai, Y., and IJ. Wijnands, "Cisco Systems' Solution for Multicast in BGP/MPLS IP VPNs", RFC 6037, October 2010.

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

[RFC6513] Rosen, E. and R. Aggarwal, "Multicast in MPLS/BGP IP VPNs", RFC 6513, February 2012.

[RFC6514] Aggarwal, R., Rosen, E., Morin, T., and Y. Rekhter, "BGP Encodings and Procedures for Multicast in MPLS/BGP IP VPNs", RFC 6514, February 2012.

8.2. Informative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

Authors' Addresses

Yisong Liu
Huawei
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: liuyisong@huawei.com

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

Email: guofeng@huawei.com