A YANG Data Model for MPLS Base
draft-ietf-mpls-base-yang-11

Abstract

This document contains a specification of the MPLS base YANG model. The MPLS base YANG model serves as a base framework for configuring and managing an MPLS switching subsystem on an MPLS-enabled router. It is expected that other MPLS YANG models (e.g. MPLS LSP Static, LDP or RSVP-TE YANG models) will augment the MPLS base YANG model.

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 March 15, 2020.

Copyright Notice

Copyright (c) 2019 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 (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Sabloki, et al. Expires March 15, 2020 [Page 1]
A core routing data model is defined in [RFC8349], and it provides a basis for the development of data models for routing protocols. The MPLS base model augments core routing data model with additional data specific to MPLS technology as described in the MPLS architecture document [RFC3031].

The MPLS base model serves as a basis for future development of MPLS data models covering more-sophisticated MPLS feature(s) and subsystem(s). The main purpose is to provide essential building blocks for the more-complicated data models involving different control-plane protocols, and advanced MPLS functions.

To this end, it is expected that the MPLS base data model will be augmented by a number of other modules developed at IETF (e.g. by TEAS and MPLS working groups).

The YANG module in this document conforms to the Network Management Datastore Architecture (NMDA) [RFC8342].
1.1. Terminology

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

The terminology for describing YANG data models is found in [RFC7950].

1.2. Acronyms and Abbreviations

MPLS: Multiprotocol Label Switching
RIB: Routing Information Base
LSP: Label Switched Path
LSR: Label Switching Router
LER: Label Edge Router
FEC: Forwarding Equivalence Class
NHLFE: Next Hop Label Forwarding Entry
ILM: Incoming Label Map

2. MPLS Base Model

This document describes the ietf-mpls YANG module that provides base components of the MPLS data model. It is expected that other MPLS YANG modules will augment the ietf-mpls base module for other MPLS extension to provision LSP(s) (e.g. MPLS Static, MPLS LDP or MPLS RSVP-TE LSP(s)).

2.1. Model Overview

This document defines a mechanism to model MPLS labeled routes as an augmentation of the routing RIB data model defined in [RFC8349] for IP prefix routes that are MPLS labeled.

The other MPLS route(s) that are non-IP prefix routes are modelled by introducing a new "mpls" address-family RIB as per recommendation.
2.2. Model Organization

Routing module +------------+ v: import
    | ietf-routing | o: augment
    +------------+

MPLS base +----------+ v: import
    | ietf-mpls | o: augment
    +----------+

MPLS Static | ietf-mpls-static@ | ietf-mpls-ldp.yang@ | ...
LSP module +-------------------+ +---------------------+

Figure 1: Relationship between MPLS modules

ietf-mpls module contains the following high-level types and groupings:

label-block-alloc-mode:

A base YANG identity for supported label block allocation mode(s).

mpls-operations-type:

An enumeration type that represents support possible MPLS operation types (impose-and-forward, pop-and-forward, pop-impose-and-forward, and pop-and-lookup)

nhlfe-role:

An enumeration type that represents the role of the NHLFE entry.

nhlfe-single-contents:

A YANG grouping that describes single NHLFE and its associated parameters as described in the MPLS architecture document [RFC3031].

nhlfe-multiple-contents:
A YANG grouping that describes a set of NHIFE(s) and their associated parameters as described in the MPLS architecture document [RFC3031].

interface-mpls-properties:

A YANG grouping that describes the properties of an MPLS interface on a device.

interfaces-mpls:

A YANG grouping that describes the list of MPLS enabled interfaces on a device.

label-block-properties:

A YANG grouping that describes the properties of an MPLS label block.

label-blocks:

A YANG grouping that describes the list of MPLS enabled interfaces on a device.

2.3. Model Tree Diagram

The MPLS base tree diagram that follows the notation defined in [RFC8340] is shown in Figure 2.

module: ietf-mpls
  augment /rt:routing:
    +++rw mpls
      +++rw ttl-propagate?  boolean
      +++rw label-blocks
        +++rw label-block* [index]
          +++rw index  string
          +++rw start-label?  rt-types:mpls-label
          +++rw end-label?  rt-types:mpls-label
          +++rw block-allocation-mode?  identityref
          +++ro free-labels-count?  yang:counter32
          +++ro inuse-labels-count?  yang:counter32
        +++rw interface* [name]
          +++rw name  if:interface-ref
          +++rw enabled?  boolean
          +++rw mtu?  uint32
  augment /rt:routing/rt:ribs/rt:rib/rt:routes/rt:route:
    +++ro mpls-enabled?  boolean
    +++ro local-label?  rt-types:mpls-label
/rt:next-hop-options/rt:simple-next-hop:
  ---ro mpls-label-stack
    +++ro entry* [id]
      ++-ro id               uint8
      ++-ro label?           rt-types:mpls-label
      ++-ro ttl?             uint8
      ++-ro traffic-class?   uint8

/rt:next-hop-options/rt:next-hop-list/rt:next-hop-list
/rt:next-hop:
  ---ro index?             string
  ---ro backup-index?      string
  ---ro loadshare?         uint16
  ---ro role?              nhlfe-role
  ---ro mpls-label-stack
    +++ro entry* [id]
      ++-ro id               uint8
      ++-ro label?           rt-types:mpls-label
      ++-ro ttl?             uint8
      ++-ro traffic-class?   uint8

augment /rt:routing/rt:ribs/rt:rib/rt:active-route/rt:input:
  +++w local-label?   rt-types:mpls-label

/rt:next-hop/rt:next-hop-options/rt:simple-next-hop:
  --- mpls-label-stack
    +++ro entry* [id]
      ++- id               uint8
      ++- label?           rt-types:mpls-label
      ++- ttl?             uint8
      ++- traffic-class?   uint8

/rt:next-hop/rt:next-hop-list/rt:next-hop:
  +++ index?             string
  +++ backup-index?      string
  +++ loadshare?         uint16
  +++ role?              nhlfe-role
  +++ mpls-label-stack
    +++ro entry* [id]
      ++- id               uint8
      ++- label?           rt-types:mpls-label
      ++- ttl?             uint8
      ++- traffic-class?   uint8

Figure 2: MPLS Base tree diagram
2.4. Model YANG Module

This section describes the "ietf-mpls" YANG module that provides base components of the MPLS data model. Other YANG module(s) may import and augment the base MPLS module to add feature specific data.

The ietf-mpls module imports the following modules:

- ietf-routing defined in [RFC8349]
- ietf-routing-types defined in [RFC8294]
- ietf-interfaces defined in [RFC8343]

<CODE BEGINS> file "ietf-mpls@2019-09-11.yang"
module ietf-mpls {
  yang-version 1.1;

  namespace "urn:ietf:params:xml:ns:yang:ietf-mpls";

  /* Replace with IANA when assigned */
  prefix "mpls";

  import ietf-routing {
    prefix "rt";
    reference "RFC8349: A YANG Data Model for Routing Management";
  }

  import ietf-routing-types {
    prefix "rt-types";
    reference "RFC8294: Common YANG Data Types for the Routing Area";
  }

  import ietf-yang-types {
    prefix "yang";
    reference "RFC6991: Common YANG Data Types";
  }

  import ietf-interfaces {
    prefix "if";
    reference "RFC8343: A YANG Data Model for Interface Management";
  }

  organization "IETF MPLS Working Group";

  contact
    "WG Web: <http://tools.ietf.org/wg/mpls>"
This YANG module defines the essential components for the management of the MPLS subsystem. The model fully conforms to the Network Management Datastore Architecture (NMDA).

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 (https://trustee.ietf.org/license-info).

This version of this YANG module is part of RFC XXXX; see the RFC itself for full legal notices.

// RFC Ed.: replace XXXX with actual RFC number and remove this note.

// RFC Ed.: update the date below with the date of RFC publication // and remove this note.

revision "2019-09-11" {
    description
        "Latest revision:
        - Addressed review comments"
        reference "RFC XXXX: A YANG Data Model for base MPLS"
}

/* Identities */
identity mpls {
    base rt:address-family;
    description
        "This identity represents the MPLS address family.";
}

identity label-block-alloc-mode {
    description
        "Base identity label-block allocation mode";
}

identity label-block-alloc-mode-manager {
    base label-block-alloc-mode;
    description
        "Label block allocation on reserved block
         is managed by label manager";
}

identity label-block-alloc-mode-application {
    base label-block-alloc-mode;
    description
        "Label block allocation on reserved block
         is managed by application";
}

/**
 * Typedefs
 */
typedef mpls-operations-type {
    type enumeration {
        enum impose-and-forward {
            description
                "Operation impose outgoing label(s) and forward to
                 next-hop";
        }
        enum pop-and-forward {
            description
                "Operation pop incoming label and forward to next-hop";
        }
        enum pop-impose-and-forward {
            description
                "Operation pop incoming label, impose one or more
                 outgoing label(s) and forward to next-hop";
        }
        enum swap-and-forward {
            description
                "Operation swap incoming label, with outgoing label and
                 forward to next-hop";
    }
enum pop-and-lookup {
    description "Operation pop incoming label and perform a lookup";
}

description "MPLS operations types";

typedef nhlfe-role {
    type enumeration {
        enum PRIMARY {
            description "Next-hop acts as primary traffic carrying";
        }
        enum BACKUP {
            description "Next-hop acts as backup";
        }
        enum PRIMARY_AND_BACKUP {
            description "Next-hop acts as primary and backup simultaneously";
        }
    }
    description "The next-hop role";
}

grouping nhlfe-single-contents {
    description "MPLS simple NHLFE contents";
    uses rt-types:mpls-label-stack;
}

grouping nhlfe-multiple-contents {
    description "MPLS NHLFE contents";
    leaf index {
        type string;
        description "A user-specified identifier utilised to uniquely reference the next-hop entry in the next-hop list. The value of this index has no semantic meaning other than for referencing the entry.";
    }

    leaf backup-index {
        type string;
        description
    }
}
"A user-specified identifier utilised to uniquely reference the backup next-hop entry in the NHLFE list. The value of this index has no semantic meaning other than for referencing the entry."

leaf loadshare {
  type uint16;
  description
  "This value is used to compute a loadshare to perform un-equal load balancing when multiple outgoing next-hop(s) are specified. A share is computed as a ratio of this number to the total under all next-hops(s).";
}

leaf role {
  type nhlfe-role;
  description "NHLFE role";
}

uses nhlfe-single-contents;

grouping interface-mpls-properties {
  description "MPLS interface contents grouping";
  leaf enabled {
    type boolean;
    description
    "'true' if mpls encapsulation is enabled on the interface. 'false' if mpls encapsulation is enabled on the interface."
  }
  leaf mtu {
    type uint32;
    description
    "MPLS Maximum Transmission Unit (MTU) in bytes";
  }
}

grouping interfaces-mpls {
  description "List of MPLS interfaces";
  list interface {
    key "name";
    description "List of MPLS interfaces";
    leaf name {
      type if:interface-ref;
      description
      "The name of a configured MPLS interface";
    }
  }
}
uses interface-mpls-properties;
}
}

grouping label-block-properties {
  description "Label-block configuration items";
  leaf index {
    type string;
    description "A user-specified identifier utilised to uniquely reference an MPLS label block";
  }
  leaf start-label {
    type rt-types:mpls-label;
    must '. >= ../end-label' {
      error-message
      "The start-label must be less than or equal " +
      "to end-label";
    }
    description "Label-block start";
  }
  leaf end-label {
    type rt-types:mpls-label;
    must '. <= ../start-label' {
      error-message
      "The end-label must be greater than or equal " +
      "to start-label";
    }
    description "Label-block end";
  }
  leaf block-allocation-mode {
    type identityref {
      base label-block-alloc-mode;
    }
    description "Label-block allocation mode";
  }
}

grouping label-block_state {
  description "Label-block state items";
  leaf free-labels-count {
    when ".../block-allocation-mode = " +
    "'label-block-alloc-mode-manager'";
    type yang:counter32;
    config false;
    description "Label-block free labels count";
  }
}
leaf inuse-labels-count {
    when ".//block-allocation-mode = " + 
    "label-block-alloc-mode-manager";
    type yang:counter32;
    config false;
    description "Label-block inuse labels count";
}

grouping globals {
    description "MPLS global configuration grouping";
    leaf ttl-propagate {
        type boolean;
        default 'true';
        description "Propagate TTL between IP and MPLS";
    }
}

grouping label-blocks {
    description "Label-block allocation grouping";
    container label-blocks {
        description "Label-block allocation container";
        list label-block {
            key index;
            unique "start-label end-label";
            description "List of MPLS label-blocks";
            uses label-block-properties;
            uses label-block_state;
        }
    }
}

augment "/rt:routing" {
    description "MPLS augmentation.";
    container mpls {
        description "MPLS container, to be used as an augmentation target node
                    other MPLS sub-features config, e.g. MPLS static LSP, MPLS
                    LDP LSPs, and Traffic Engineering MPLS LSP Tunnels, etc.";
                    uses globals;
                    uses label-blocks;
                    uses interfaces-mpls;
    }
}

/* MPLS routes augmentation */
augment "/rt:routing/rt:ribs/rt:rib/rt:routes/rt:route" {
    description "This is augmentation for all MPLS routes.";
leaf mpls-enabled {
  type boolean;
  default 'false';
  description "Indicates whether MPLS is enabled for this route";
}
leaf local-label {
  when "./mpls-enabled = 'true'";
  type rt-types:mpls-label;
  description "MPLS local label associated with the route.";
}

  + "rt:next-hop/rt:next-hop-options/rt:simple-next-hop" {
  description "Augment 'simple-next-hop' case in IP unicast routes.";
  uses nhlfe-single-contents {
    when "/rt:routing/rt:ribs/rt:rib/rt:routes/rt:route" + 
    "/mpls:mpls-enabled = 'true'";
  }
}

  + "rt:next-hop/rt:next-hop-options/rt:next-hop-list/" + 
  "rt:next-hop-list/rt:next-hop" {
  description "This leaf augments the 'next-hop-list' case of IP unicast routes.";
  uses nhlfe-multiple-contents {
    when "/rt:routing/rt:ribs/rt:rib/rt:routes/rt:route" + 
    "/mpls:mpls-enabled = 'true'";
  }
}

augment "/rt:routing/rt:ribs/rt:rib/rt:active-route/rt:input" {
  description "Input MPLS augmentation for the 'active-route' action statement.";
  leaf local-label {
    type rt-types:mpls-label;
    description "MPLS local label.";
  }
}
augment "/rt:routing/rt:ribs/rt:rib/rt:active-route/"
   + "rt:output/rt:route/"
   + "rt:next-hop/rt:next-hop-options/rt:simple-next-hop" {
   description
   "Output MPLS augmentation for the 'active-route' action statement.";
   uses nhlfe-single-contents;
}

augment "/rt:routing/rt:ribs/rt:rib/rt:active-route/"
   + "rt:output/rt:route/"
   + "rt:next-hop/rt:next-hop-options/rt:next-hop-list/"
   + "rt:next-hop-list/rt:next-hop" {
   description
   "Output MPLS augmentation for the 'active-route' action statement.";
   uses nhlfe-multiple-contents;
}

Figure 3: MPLS base YANG module

3. IANA Considerations

This document registers the following URIs in the IETF XML registry
[RFC3688]. Following the format in [RFC3688], the following
registration is requested to be made.

   Registrant Contact: The MPLS WG of the IETF.
   XML: N/A, the requested URI is an XML namespace.

This document registers a YANG module in the YANG Module Names
registry [RFC6020].

   name:    ietf-mpls
   prefix:  ietf-mpls
   // RFC Ed.: replace XXXX with RFC number and remove this note
   reference: RFCXXXX
4. Security Considerations

The YANG modules specified in this document define a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

The NETCONF access control model [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

Some of the readable data nodes in these YANG modules may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

/routing/ribs/rib/active-route/output/route: this path is augmented by additional MPLS leaf(s) defined in this model. Access to this information may disclose the per prefix and/or other information.

/routing/ribs/rib/active-route/output/route/rt:next-hop/rt:next-hop-options/rt:simple-next-hop: this path is augmented by additional MPLS leaf(s) defined in this model. Access to this information may disclose the next-hop or path per prefix and/or other information.

5. Acknowledgement

The authors would like to thank the members of the multi-vendor YANG design team who are involved in the definition of this model.

6. Contributors

Igor Bryskin
Huawei Technologies
email: Igor.Bryskin@huawei.com

Himanshu Shah
Ciena
email: hshah@ciena.com
7. References

7.1. Normative References


7.2. Informative References


Authors' Addresses

Tarek Saad
Juniper Networks
Email: tsaad@juniper.net

Kamran Raza
Cisco Systems Inc
Email: skraza@cisco.com

Rakesh Gandhi
Cisco Systems Inc
Email: rgandhi@cisco.com
Xufeng Liu
Volta Networks

Email: xufeng.liu.ietf@gmail.com

Vishnu Pavan Beeram
Juniper Networks

Email: vbeeram@juniper.net