MPLS Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 18, 2016

T. Saad
K. Raza
R. Gandhi
Cisco Systems Inc
X. Liu
Ericsson
V. Beeram
Juniper Networks
H. Shah
Ciena
X. Chen
Huawei Technologies
R. Jones
Brocade
B. Wen
Comcast

October 16, 2015

# A YANG Data Model for MPLS Base and Static LSPs
# draft-saad-mpls-static-yang-00

Abstract

This document contains a specification of two YANG modules, the MPLS base, and Static LSP YANG modules. The MPLS base YANG module serves as a base framework for configuring and managing an MPLS switching subsystem. The MPLS Static LSP module augments the MPLS base YANG module with specific data to configure and manage MPLS Static LSP(s). It is expected that other MPLS YANG modules for MPLS technology YANG modules (e.g. MPLS LDP or MPLS RSVP-TE) will also augment the MPLS YANG base model.

This Internet-Draft will expire on April 18, 2016.



Table of Contents



## 1. Introduction

AAA

### 1.1. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in BCP 14, RFC 2119 [RFC2119].

The following terms are defined in [RFC6020]:

- augment,
- configuration data,
- data model,
- data node,
- feature,
- mandatory node,
- module,
- schema tree,
- state data,
- RPC operation.

## 2. Introduction

A core routing data model is defined in [I-D.ietf-netmod-routing-cfg], and it proposes a basis for the development of data models for routing protocols. The MPLS base model augments this model with additional data specific to MPLS switching. The interface data model is defined in [RFC7223] and is used for referencing interface from the MPLS base model.

This document contains a specification of the following YANG modules:

- Module "ietf-mpls" provides base components of the MPLS data model.
- Module "ietf-mpls-static" that augments "ietf-mpls" with additional data specific to configuration and management of Static LSPs.

Note, it is expected that other MPLS YANG models to augment the "ietf-mpls" base model to define specific data to other MPLS technologies (e.g. MPLS LDP or MPLS RSVP-TE).

### 2.1. Overview

The MPLS base YANG module augments the "routing/routing-instance/ routing- protocols/routing-protocol" path for the "rw" branch, and augments path "routing-state/routing-instance/routing- protocols/ routing-protocol" for the "ro" branch of the ietf-routing module.

The approach described in [I-D.openconfig-netmod-opstate] is adopted to represent data pertaining to configuration intended, applied state and derived state data elements. Each container in the model holds a "config" and "state" sub-container. The "config" sub-container is used to represent the intended configurable parameters, and the state sub-container is used to represent both the applied configurable parameters and any derived state, such as counters or statistical information.

### 2.1.1. MPLS Base Tree Diagram

The MPLS base tree diagram is shown in Figure 1.

```
module: ietf-mpls
augment /rt:routing/rt:routing-instance:
   +--rw mpls
      +--rw interface* [name]
         +--rw name      if:interface-ref
         +--rw config
         |  +--rw enabled?   boolean
         +--ro state
            +--ro enabled?   boolean
augment /rt:routing-state/rt:routing-instance:
   +--ro mpls
```

Figure 1: MPLS Base tree diagram

### 2.1.2. MPLS Static LSP Tree Diagram

The MPLS Static LSP tree diagram is shown in Figure 2.

```
module: ietf-mpls-static
augment /rt:routing/rt:routing-instance/mpls:mpls:
   +--rw static-lsps
      +--rw static-lsp* [name]
         +--rw name      string
         +--rw config
         |  +--rw in-segment
         |  |  +--rw (type)?
         |  |     +--:(ip-prefix)
         |  |     |  +--rw ip-prefix?        inet:ip-prefix
         |  |     +--:(mpls-label)
         |  |        +--rw incoming-label?   mpls:mpls-label
         |  +--rw operation?            enumeration
         |  +--rw (out-segment)?
         |     +--:(simple-path)
         |     |  +--rw next-hop?             inet:ip-address
         |     |  +--rw outgoing-label?       mpls:mpls-label
```

```
            |     |  +--rw outgoing-interface?   if:interface-ref
            |     +--:(path-list)
            |        +--rw paths* [path-index]
            |           +--rw path-index            uint32
            |           +--rw backup-path-index?    uint32
            |           +--rw next-hop?             inet:ip-address
            |           +--rw outgoing-labels* [index]
            |           |  +--rw index    uint32
            |           |  +--rw label?   mpls:mpls-label
            |           +--rw outgoing-interface?   if:interface-ref
            |           +--rw loadshare?            mpls:percent
            |           +--rw role?                 enumeration
            +--ro state
               +--ro in-segment
               |  +--ro (type)?
               |     +--:(ip-prefix)
               |     |  +--ro ip-prefix?        inet:ip-prefix
               |     +--:(mpls-label)
               |        +--ro incoming-label?   mpls:mpls-label
               +--ro operation?            enumeration
               +--ro (out-segment)?
                  +--:(simple-path)
                  |  +--ro next-hop?             inet:ip-address
                  |  +--ro outgoing-label?       mpls:mpls-label
                  |  +--ro outgoing-interface?   if:interface-ref
                  +--:(path-list)
                     +--ro paths* [path-index]
                        +--ro path-index            uint32
                        +--ro backup-path-index?    uint32
                        +--ro next-hop?             inet:ip-address
                        +--ro outgoing-labels* [index]
                        |  +--ro index    uint32
                        |  +--ro label?   mpls:mpls-label
                        +--ro outgoing-interface?   if:interface-ref
                        +--ro loadshare?            mpls:percent
                        +--ro role?                 enumeration
```

Figure 2: MPLS Static LSP tree diagram

## 2.2. YANG Modules

### 2.2.1. MPLS Base Module

```
<CODE BEGINS>file "ietf-mpls@2015-10-16.yang"

module ietf-mpls {

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

```
  prefix "mpls";

  import ietf-routing {
    prefix "rt";
  }

  import ietf-interfaces {
    prefix "if";
  }

  organization "TBD";

  contact "TBD";

  description
    "This YANG module defines the essential components for the
     management of the MPLS subsystem.";

  revision "2015-10-16" {
    description
      "Initial revision";
    reference "RFC 3031: A YANG Data Model for base MPLS";
  }

  typedef mpls-label {
    type uint32 {
      range "0..1048575";
    }
    description
      "The MPLS label range";
  }

  typedef percent {
    type uint16 {
      range "0 .. 100";
    }
    description "Percentage";
  }

  grouping interface-mpls {
    description "MPLS interface properties grouping";
    leaf enabled {
      type boolean;
      description
        "'true' if mpls encapsulation is enabled on the
         interface. 'false' if mpls encapsulation is enabled
         on the interface.";
    }
```

```
    }

    augment "/rt:routing/rt:routing-instance" {
      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 Trafic Engineering MPLS LSP Tunnels, etc.";

        list interface {
          key "name";
          description "List of MPLS interfaces";
          leaf name {
            type if:interface-ref;
            description
              "The name of a configured MPLS interface";
          }
          container config {
            description "Holds intended configuration";
            uses interface-mpls;
          }
          container state {
            config false;
            description "Holds inuse configuration";
            uses interface-mpls;
          }
        }
      }
    }

    augment "/rt:routing-state/rt:routing-instance" {
      description "MPLS augmentation.";
      container mpls {
        config false;
        description
          "MPLS container, to be used as an augmentation target node
           other MPLS sub-features state";
      }
    }
  }

  <CODE ENDS>
```

Figure 3: MPLS base YANG module

### 2.2.2. MPLS Static LSP YANG Module

```
<CODE BEGINS>file "ietf-mpls@2015-10-16.yang"

module ietf-mpls-static {

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

  prefix "mpls-static";

  import ietf-mpls {
    prefix mpls;
  }

  import ietf-routing {
    prefix "rt";
  }

  import ietf-inet-types {
    prefix inet;
  }

  import ietf-interfaces {
    prefix "if";
  }

  organization "TBD";

  contact "TBD";

  description
     "This YANG module augments the 'ietf-routing' module with basic
     configuration and operational state data for MPLS static";

  revision "2015-10-16" {
    description
      "Initial revision";
    reference
      "RFC 3031: A YANG Data Model for MPLS Static";
  }

  grouping path-basic_config {
    description "common definitions for statics";

    leaf next-hop {
      type inet:ip-address;
      description "next hop IP address for the LSP";
    }
```

```
        leaf outgoing-label {
          type mpls:mpls-label;
          description
            "label value to push at the current hop for the
             LSP";
        }

        leaf outgoing-interface {
          type if:interface-ref;
          description
            "The outgoing interface";
        }

      }

      grouping path-properties_config {
        description
          "MPLS path properties";
        leaf path-index {
          type uint32;
          description
            "Path identifier";
        }

        leaf backup-path-index {
          type uint32;
          description
            "Backup path identifier";
        }

        leaf next-hop {
          type inet:ip-address;
          description
            "The address of the next-hop";
        }

        list outgoing-labels {
          key index;
          description
            "The outgoing MPLS labels to impose";
          leaf index {
            type uint32;
            description
              "The index of the label, lower indices are closer to
               the top";
          }
          leaf label {
            type mpls:mpls-label;
```

```
          description
            "The MPLS label value";
        }
      }

      leaf outgoing-interface {
        type if:interface-ref;
        description
          "The outgoing interface";
      }

      leaf loadshare {
        type mpls:percent;
        description
          "The percentage of total load tto carry on this path";
      }

      leaf role {
        type enumeration {
          enum PRIMARY {
            description
              "Path as primary traffic carrying";
          }
          enum BACKUP {
            description
              "Path acts as backup";
          }
          enum PRIMARY_AND_BACKUP {
            description
              "Path acts as primary and backup simultaneously";
          }
        }
        description
          "The MPLS path role";
      }
    }

    grouping static-lsp_config {
      description "common definitions for static LSPs";

      container in-segment {
        description
          "MPLS incoming segment";
        choice type {
          description
            "Basic FEC choice";
          case ip-prefix {
            leaf ip-prefix {
```

```
            type inet:ip-prefix;
            description "An IP prefix";
          }
        }
        case mpls-label {
          leaf incoming-label {
            type mpls:mpls-label;
            description "label value on the incoming packet";
          }
        }
      }
    }

    leaf operation {
      type enumeration {
        enum impose-and-forward {
          description
            "Operation impose outgoing label(s) and forward to
             next-hop";
        }
        enum pop-and-forward {
          description
            "Operation pop outgoing 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
        "The MPLS operation to be executed on the incoming packet";
    }

    choice out-segment {
      description "The MPLS out-segment type choice";
      case simple-path {
        uses path-basic_config;
      }
```

```
        case path-list {
          list paths {
            key path-index;
            description
              "The list of MPLS paths associated with the FEC";
            uses path-properties_config;
          }
        }
      }
    }

    grouping static-lsp {
      description "grouping for top level list of static LSPs";
      container config {
        description
          "Holds the intended configuration";
        uses static-lsp_config;
      }
      container state {
        config false;
        description
          "Holds the state and inuse configuration";
        uses static-lsp_config;
      }
    }

    augment "/rt:routing/rt:routing-instance/mpls:mpls" {
      description "Augmentations for MPLS Static LSPs";
      container static-lsps {
        description
          "Statically configured LSPs, without dynamic signaling";
        list static-lsp {
          key name;
          description "list of defined static LSPs";

          leaf name {
            type string;
            description "name to identify the LSP";
          }
          uses static-lsp;
        }
      }
    }
  }

  <CODE ENDS>
```

Figure 4: MPLS Static LSP 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.

URI: urn:ietf:params:xml:ns:yang:ietf-mpls XML: N/A, the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-mpls-static 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 namespace: urn:ietf:params:xml:ns:yang:ietf-mpls prefix: ietf-mpls reference: RFC3031

name: ietf-mpls-static namespace: urn:ietf:params:xml:ns:yang:ietf-mpls-static prefix: ietf-mpls-static reference: RFC3031

## 4. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The NETCONF access control model [RFC6536] provides means to restrict access for particular NETCONF users to a pre-configured subset of all available NETCONF protocol operations and content.

There are a number of data nodes defined in the YANG module which are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., <edit-config>) to these data nodes without proper protection can have a negative effect on network operations.

## 5. References

### 5.1. 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-20 (work in progress), October 2015.

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

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

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

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

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

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

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

## 5.2. Informative References

[I-D.openconfig-netmod-opstate] Shakir, R., Shaikh, A., and M. Hines, "Consistent Modeling of Operational State Data in YANG", draft-openconfig-netmod-opstate-01 (work in progress), July 2015.

Authors' Addresses

Tarek Saad
Cisco Systems Inc

Email: tsaad@cisco.com

Kamran Raza
Cisco Systems Inc

Email: skraza@cisco.com

Rakesh Gandhi
Cisco Systems Inc

Email: rgandhi@cisco.com

Xufeng Liu
Ericsson

Email: xufeng.liu@ericsson.com

Vishnu Pavan Beeram
Juniper Networks

Email: vbeeram@juniper.net

Himanshu Shah
Ciena

Email: tsaad@cisco.com

Xia Chen
Huawei Technologies

Email: jescia.chenxia@huawei.com

Raqib Jones
Brocade

Email: raqib@Brocade.com

Bin Wen
Comcast

Email: Bin_Wen@cable.comcast.com