MPLS Working Group
Internet-Draft
Intended Status: Standards Track
Expires: April 30, 2015

Rakesh Gandhi
Tarek Saad
Robert Sawaya
Cisco Systems, Inc.

October 27, 2014

YANG Data Model for MPLS Traffic Engineering Tunnels and Links
draft-gandhi-mpls-te-yang-model-01

Abstract

This document defines YANG data model for the management of Multi-Protocol Label Switching Traffic Engineering (MPLS-TE) tunnels, Label Switched Paths (LSPs) and links. The data model covers configuration data, operational state data, execution commands and event notifications.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

This document defines YANG [RFC6020] data model for the management of Multi-Protocol Label Switching Traffic Engineering (MPLS-TE) [RFC3209] tunnels and links. Resource Reservation Protocol (RSVP) [RFC2205] signaled MPLS-TE paths can be represented as tunnels at the head-end Label Switching Router (LSR), and as Label Switched Paths (LSPs) at the head-end, mid-point and tail-end LSRs.

The data model defined in this document includes configuration data, operational state data (status information and counters), execution requests using RPCs (Remote Procedure Calls), and event notifications pertaining to MPLS-TE tunnels, LSPs and MPLS-TE enabled links, as well as system-wide global MPLS-TE properties that relate to the behavior and operation of the TE enabled LSR node.

Further modules augmenting this data model with advanced features can be handled in a future revision or a separate document.

2. Terminology and Notation

2.1. Terminology

The keywords "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].

2.2. Prefixes in Data Node Names

In this document, names of data nodes and other data model objects are often used without a prefix, as long as it is clear from the context in which YANG module each name is defined. Otherwise, names are prefixed using the standard prefix associated with the corresponding YANG module, as shown in Table 1.

Prefix	YANG module	Reference
yang	ietf-yang-types	[RFC6991]
inet	ietf-inet-types	[RFC6991]

Table 1: Prefixes and corresponding YANG modules

3. Objectives

This section outlines some of the design objectives for the model:

- o In case of existing implementations, it needs to map the data model defined in this document to their proprietary native data model. To facilitate such mappings, the data model should be simple.

- o The data model should be suitable for new implementations to use as is, without requiring a mapping to a different native model.

- o Mapping to the MPLS-TE MIB Module should be clear.

- o The data model should include read-only counters in order to gather statistics for sent and received octets and packets, received packets with errors, and packets that could not be sent due to errors.

- o It should be straight forward to augment the base data model for advanced MPLS-TE features.

4. MPLS-TE Data Models Overview

MPLS-TE YANG data models are defined for various management components including configuration, operational state, execution commands and event notifications. Following sections provide overview and some selective examples of these management components for global MPLS-TE, MPLS-TE tunnels, MPLS-TE LSPs and MPLS-TE enabled links.

4.1. Global MPLS-TE Data Model Overview

This module defines YANG data model for configuration data, operational state data, execution commands and event notifications globally for MPLS-TE features.

1. Global MPLS-TE configuration data model: The global MPLS-TE configuration data model is a read-write YANG data model that controls the LSR behavior system-wide. Examples of such configuration items for global MPLS-TE are:

   - o Auto-tunnel backup: controls and manages the automatic creation of fast reroute backup tunnels for protected MPLS-TE enabled links.
   - o Auto-tunnel mesh-group: controls and manages the automatic creation of tunnels for mesh-groups.
   - o Auto-tunnel one-hop: controls and manages the automatic

creation of 1-hop tunnels on MPLS-TE enabled links.
- Auto-bandwidth parameters: controls and manages the

  auto-bandwidth specific system-wide properties.
- System-wide Point-to-Multipoint (P2MP) TE parameters
- Names for SRLG values
- Names for link (extended) administrative groups (AG, EAG)
- MPLS-TE Diff-Serve classes: controls and manages the Diff-Serve TE (DS-TE) model and TE-class maps
- System-wide capabilities for TE LSP re-optimization e.g.
  - Re-optimization times (periodic interval, installation, cleanup)
- System-wide capabilities for MPLS TE link flooding e.g.
  - Periodic flooding interval, bandwidth threshold values.
- MPLS-TE tunnel templates: These are templates that can be used to instantiate tunnels and LSPs with identical configuration properties.
- System-wide capabilities that affect the originating, traversing and terminating LSPs. For example:
  - CSPF metric (TE or IGP)
  - Handling for over-loaded nodes
  - (Soft) preemption parameters
  - Path protection parameters at the head-end LSR
  - Fast reroute parameters

2. Global MPLS-TE state data model: The global MPLS-TE state data model is a read-only YANG data model. This module defines system-wide operational data for various MPLS-TE features. Examples of such system-wide MPLS-TE states are:

- Global statistics (signaling, admission, preemption, flooding)
- Global counters (number of tunnels/LSPs/links)

3. Global MPLS-TE execution data model: The global MPLS-TE execution model facilitates issuing commands to an LSR node and optionally returning responses. This model uses RPC operations and contains optional read-only input and output data. Examples of such global MPLS-TE commands are:

- Clear global MPLS-TE statistics of various features

4. Global MPLS-TE events notification data model: The global MPLS-TE events notification model uses configuration data for registration. Node notifies the registered events to the server using notification messages. Notifications carry read-only data in the messages. Example of such global MPLS-TE events are:

- Backup tunnel FRR active and not-active state transition events

4.2. MPLS-TE Tunnel Data Model Overview

This module defines configuration data, operational state data, execution commands and event notifications for MPLS-TE tunnels and is applicable to head-end LSRs.

1. MPLS-TE tunnels configuration data model: The configuration data model is a read-write YANG data model. This module defines configuration items for the MPLS-TE tunnels. Examples of such configuration items are:

   - Name
   - Admin-state
   - Tunnel-type (such as P2P, P2MP)
   - Routing properties (such as auto-route announce, forwarding adjacency)
   - Forwarding properties (such as traffic class based tunnel selection)
   - Static route information
   - LDP over tunnel parameters
   - Quality of Service (QoS) policy parameters

2. MPLS-TE tunnel state data model: The MPLS-TE tunnel state data model is a read-only YANG data model. This module defines operational state data for MPLS-TE tunnels at the head-end LSRs. Examples of such MPLS-TE tunnel states are:

   - Name
   - Tunnel creation information (time and trigger: static-configuration/auto-tunnel)
   - State information (Up/Down: when and reason)
   - Traffic counters
   - History of events

3. MPLS-TE tunnel execution data model: The execution model facilitates issuing commands to an LSR node and optionally returning responses. This model uses RPC operations and contains optional read-only input and output data. Example commands for MPLS-TE tunnels are:

   - Clear statistics for all or for individual tunnels

4. MPLS-TE tunnel events notification data model: The notification model uses configuration data for registration. Node notifies the registered events to the server using notification messages. Notifications carry read-only data in the messages. Example events for MPLS-TE tunnels are:

- Tunnel creation and deletion events
- Tunnel state transition events

4.3. MPLS-TE Tunnel LSP Data Model Overview

This module defines configuration data, operational state data, execution commands and event notifications for MPLS-TE tunnel LSPs and is applicable to head-end, mid-point and tail-end LSRs.

1. MPLS-TE tunnel LSP configuration data model: The configuration data model is a read-write YANG data model component. This module defines configuration items for the MPLS-TE tunnel LSP properties. Examples of such MPLS-TE tunnel LSP configuration items are:

   - Name
   - Signaling properties (such as bandwidth, class-type, set-up and hold priorities)
   - Path-computation parameters (dynamic path, explicit path, cost-limit, hop-limit, metric type, affinity parameters (colors))

2. MPLS-TE tunnel LSP state data model: The MPLS-TE tunnel LSP state data model is a read-only YANG data model. This model defines the operational state data for MPLS-TE tunnel LSPs for head-end, mid-point and tail-end LSRs. Example states data for MPLS-TE tunnel LSPs are:

   - Name
   - LSP creation information (time)
   - State information (Up/Down: when and reason)
   - MPLS-TE attribute-set name
   - Signaling information (Explicit Route Object, Record Route Object, bandwidth, egress and ingress links)
   - FRR information (status, type of protection, backup tunnel)
   - Soft preemption properties
   - Path protection properties
   - Statistics
   - History of events

3. MPLS-TE tunnel LSP execution data model: The execution model facilitates issuing commands to an LSR node and optionally returning responses. This model uses RPC operations and contains optional read-only input and output data. Examples of such commands for MPLS-TE tunnel LSPs are:

   - Trigger re-optimization on all or on individual LSP
   - Trigger path protection switchover on an individual LSP
   - Trigger LSP path switchover on an individual LSP

   - Clear TE statistics for all or for individual LSPs

4. MPLS-TE tunnel LSP events notification data model: The notification model uses configuration data for registration. Node notifies the registered events to the server using notification messages. Notifications carry read-only data in the messages. Examples of such events for MPLS-TE LSPs are:

   - LSP creation and deletion events
   - LSP state transition events
   - LSP re-optimization including trigger reason
   - Fast Reroute (protection availability, activation) events
   - LSP signaling events
   - Auto-bandwidth changes
   - Path protection events
   - (Soft) Preemption events

4.4. MPLS-TE Link Data Model Overview

This module defines configuration data, operational state data, execution commands and event notifications for MPLS-TE enabled links on an LSR.

1. MPLS-TE link configuration data model: The configuration data model is a read-write YANG data model component. This model defines configuration items for MPLS-TE enabled links used to advertise in TE topology database. Examples of such configuration items for MPLS-TE enabled links are:

   - Name
   - Maximum reservable bandwidth, bandwidth constraints (BC) values
   - Diff-serve model type (e.g. Russian Doll Model)
   - (Extended) Administrative groups (AGs and EAGs)
   - SRLG values
   - TE metric value
   - Flooding parameters
     - Flooding intervals and threshold values
   - RSVP Parameters
     - RSVP authentication parameters
     - RSVP refresh reduction parameters
     - RSVP hello parameters
     - RSVP graceful restart (GR)
   - Fast reroute backup tunnel properties (such as static, auto-tunnel)

2. MPLS-TE link state data model: The MPLS-TE link state model is a read-only YANG data model. This model defines operational state data

for MPLS-TE enabled links for an LSR node. Examples of such state data for MPLS-TE links are:

- o Name
- o State information (UP/Down: when and reason)
- o IGP information
  - o IGP neighbor
  - o IGP metric
- o Bandwidth information: maximum bandwidth, reserved bandwidth at different priorities, available bandwidth at different priorities and for each class-type (CT)
- o List of admitted LSPs (name, bandwidth value and pool, time, priority)
- o Statistics: state counters, flooding counters, admission counters (accepted/rejected), preemption counters
- o History of events

3. MPLS-TE link execution data model: The execution model facilitates issuing commands to an LSR node and optionally returning responses. This model uses RPC operations and contains optional read-only input and output data. Examples of such commands for MPLS-TE links are:

- o Clear TE statistics for all or for individual links
- o Trigger immediate flooding for all TE links

4. MPLS-TE link events notification data model: The notification model uses configuration data for registration. Node notifies the registered events to the server using notification messages. Notifications carry read-only data in the messages. Example events for MPLS-TE links are as following:

- o Link creation and deletion events
- o Link state transition events
- o (Soft) preemption trigger events
- o Fast reroute activation events

5. High-level Tree Structure of MPLS-TE Data Model

The module, "ietf-mpls-te", defines the YANG data model for various management components (configuration, operational, execution and notification) within MPLS-TE. The data module includes modules for global MPLS-TE, tunnels, LSPs and links and the tree structure is organized as shown below.

The following notations are used for the data tree.

```
   <status> <flags> <name> <opts> <type>

    <status> is one of:

      +  for current
      x  for deprecated
      o  for obsolete

    <flags> is one of:
      rw  for read-write configuration data
      ro  for read-only non-configuration data
      -x  for execution rpcs
      -n  for notifications

    <name> is the name of the node

    If the node is augmented into the tree from another module, its name
   is printed as <prefix>:<name>

    <opts> is one of:
      ? for an optional leaf or node
      ! for a presence container
      * for a leaf-list or list
      Brackets [<keys>] for a list’s keys
      Curly braces {<condition>} for optional feature that make node
   conditional
      Colon : for marking case nodes
      Ellipses ("...") subtree contents not shown

    <type> is the name of the type for leafs and leaf-lists.


   module ietf-mpls-te
       +--rw tunnels-cfg!
       +--rw lsps-cfg!
       +--rw links-cfg!
       +--rw global-cfg!
       +--ro tunnels-oper
       +--ro lsps-oper
       +--ro links-oper
       +--ro global-oper
    rpcs:
       +---x tunnels-rpc
       +---x lsps-rpc
       +---x global-rpc
       +---x links-rpc
    notifications:
       +---n tunnels-notif
       +---n lsps-notif
       +---n links-notif
       +---n global-notif
```

As shown, data tree structure is organized by MPLS-TE data modules, which are global MPLS-TE, tunnels, LSPs and links.
Each of this data model module contains various management components including configuration, operation, execution and notification.

6. MPLS-TE Global Data Model Subtree Structure

This document defines the YANG data model subtree for global MPLS TE configuration, state, RPCs and notifications as follows:

```
   +--rw global-cfg!
   |  +--rw path-selection
   |  |  +--rw cost-limit?   uint32
   |  |  +--rw hop-limit?    uint32
   |  |  +--rw metric?       mtt:path-metric-type
   |  |  +--rw tiebreaker?   mtt:path-tiebreaker-type
   |  +--rw bfd!
   |  |  +--rw type?                mtt:bfd-type
   |  |  +--rw encap-mode?          mtt:bfd-encap-mode-type
   |  |  +--rw bringup-timeout?     uint32
   |  |  +--rw dampening?           uint32
   |  |  +--rw lsp-ping
   |  |  |  +--rw disable?    boolean
   |  |  |  +--rw interval?   uint32
   |  |  +--rw minimum-interval?   uint32
   |  |  +--rw multiplier?         uint32
   |  +--rw logging-events
   |  |  +--rw (type)?
   |  |     +--:(all)
   |  |     |  +--rw all!
   |  |     +--:(bfd-status)
   |  |     |  +--rw bfd-status!
   |  |     +--:(link-status)
   |  |     |  +--rw link-status!
   |  |     +--:(lsp-status)
   |  |     |  +--rw lsp-status!
   |  |     |     +--rw all?      boolean
   |  |     |     +--rw events*   mtt:lsp-status-event-type
   |  |     +--:(cspf-failure)
   |  |        +--rw cspf-failure!
   |  +--rw tunnel-templates* [name]
   |  |  +--rw name                     string
   |  |  +--rw source?                  inet:ip-address
   |  |  +--rw fast-reroute!
   |  |  |  +--rw bandwidth-protect?   boolean
   |  |  |  +--rw node-protect?        boolean
   |  |  +--rw record-route?           boolean
   |  |  +--rw signaled-name?          string
   |  |  +--rw priority
   |  |  |  +--rw setup?   uint8
   |  |  |  +--rw hold?    uint8
   |  |  +--rw soft-preemption?        boolean
   |  |  +--rw signaled-bandwidth
   |  |  |  +--rw type?    mtt:bandwidth-type
   |  |  |  +--rw value?   uint32
   |  |  +--rw (style)?
   |  |  |  +--:(affinity-hex)
   |  |  |  |  +--rw affinity-hex
   |  |  |  |     +--rw value?   uint32
   |  |  |  |     +--rw mask?    uint32
   |  |  |  +--:(affinity-name)
   |  |  |     +--rw affinity-name
   |  |  |        +--rw constraints* [action]
   |  |  |           +--rw action        mtt:affinity-action-type
   |  |  |           +--rw constraint
   |  |  |              +--rw affinity-list* [name]
   |  |  |                 +--rw name    string
   |  |  +--rw logging-events
   |  |  |  +--rw (type)?
   |  |  |     +--:(all)
   |  |  |     |  +--rw all!
   |  |  |     +--:(bfd-status)
   |  |  |     |  +--rw bfd-status!
   |  |  |     +--:(link-status)
   |  |  |     |  +--rw link-status!
   |  |  |     +--:(lsp-status)
   |  |  |     |  +--rw lsp-status!
   |  |  |     |     +--rw all?      boolean
   |  |  |     |     +--rw events*   mtt:lsp-status-event-type
   |  |  |     +--:(cspf-failure)
   |  |  |        +--rw cspf-failure!
   |  |  +--rw auto-bandwidth!
   |  |  |  +--rw overflow-threshold?      uint32
   |  |  |  +--rw overflow-limit?          uint8
   |  |  |  +--rw underflow-threshold?     uint32
   |  |  |  +--rw underflow-limit?         uint8
   |  |  |  +--rw collect-only?            boolean
   |  |  |  +--rw application-frequency?   uint32
   |  |  |  +--rw bandwidth-limit
   |  |  |     +--rw min-limit?   uint32
   |  |  |     +--rw max-limit?   uint32
   |  |  +--rw (routing-proprties)?
   |  |  |  +--:(autoroute)
   |  |  |  |  +--rw autoroute!
   |  |  |  |     +--rw routing-afs*       inet:ip-version
   |  |  |  |     +--rw (metric-type)?
   |  |  |  |        +--:(metric)
   |  |  |  |        |  +--rw metric?            uint32
   |  |  |  |        +--:(relative-metric)
   |  |  |  |        |  +--rw relative-metric?   uint32
   |  |  |  |        +--:(absolute-metric)
   |  |  |  |           +--rw absolute-metric?   uint32
   |  |  |  +--:(forwarding-adjacency)
   |  |  |     +--rw forwarding-adjacency!
   |  |  |        +--rw holdtime?      uint32
   |  |  |        +--rw routing-afs*   inet:ip-version
   |  |  +--rw (forwarding-property)?
   |  |     +--:(forwarding-class)
   |  |     |  +--rw forwarding-class
   |  |     |     +--rw class?   uint8
   |  |     +--:(forwarding-group)
   |  |        +--rw forwarding-group
   |  |           +--rw classes*   uint8
   |  +--rw auto-tunnel-mesh* [mesh-group]
   |  |  +--rw mesh-group       uint32
   |  |  +--rw template-name?   string
   |  |  +--rw one-hop-only?    boolean
   |  +--rw template-name?           string
   |  +--rw nhop-only?               boolean
   |  +--rw load-share-properties
   |  |  +--rw unequal!
   |  |     +--rw bandwidth-based?   boolean
   |  +--rw global-timers
   |  |  +--rw lsp-bw-hold-delay?          uint32
   |  |  +--rw lsp-preemption-delay?       uint32
   |  |  +--rw topology-holddown-sigerr?   uint32
   |  +--rw topology-flooding!
   |  |  +--rw flooding-interval?     uint32
   |  |  +--rw flooding-thresholds
   |  |     +--rw (type)?
   |  |        +--:(single-step)
   |  |        |  +--rw up-step?      uint8
   |  |        |  +--rw down-step?    uint8
   |  |        +--:(multiple-steps)
   |  |           +--rw up-steps* [value]
   |  |           |  +--rw value    uint8
   |  |           +--rw down-steps* [value]
   |  |              +--rw value    uint8
   |  +--rw global-reoptimization!
   |  |  +--rw interval?            uint32
   |  |  +--rw installation-delay?   uint32
   |  |  +--rw cleanup-delay?        uint32
   |  +--rw softpreemption-te!
   |  |  +--rw softpremption-timeout?   uint32
   |  +--rw affinity-maps
   |  |  +--rw affinity-maps* [name]
   |  |     +--rw name         string
   |  |     +--rw (type)?
   |  |        +--:(value)
   |  |        |  +--rw value?       uint32
   |  |        +--:(bit-index)
   |  |           +--rw bit-index?   uint32
   |  +--rw srlg-maps
   |  |  +--rw srlg-maps* [name]
   |  |     +--rw name            string
   |  |     +--rw value?          uint32
   |  |     +--rw admin-weight?   uint32
   |  +--rw ds-te!
   |  |  +--rw bc-model?   mtt:bc-model-type
   |  |  +--rw te-class* [class-index]
   |  |     +--rw class-index    uint8
   |  |     +--rw priority?      uint8
   |  |     +--rw bc-value?      uint8
   |  +--rw auto-bw!
   |  |  +--rw stats-collection-interval?   uint32
   |  +--rw tail-signaling
   |  |  +--rw advertise-explicit-null?   boolean

   +--ro global-state
   |   +--ro tunnel-templates-state
   |   |  +--ro tunnel-template*  [name]
   |   |  +--ro name                    string

   +---x global-rpcs
   |   +---x te-global-rpcs-to-be-defined

   +---n global-notifications
   |   +---n te-global-notifications-to-be-defined
```

6.1. MPLS-TE Global Tunnel-template Lists

The data model for tunnel-templates presented in this document uses a flat list of tunnel-template(s). Each tunnel-template in the list is identified by its name. A tunnel-template is a configuration template that can be used to instantiate tunnels and LSPs with identical properties.

There is one list for MPLS-TE tunnel-template configurations ("/global-cfg/tunnel-templates") and a separate list for operational state data ("/global-state/tunnel-templates").

7. MPLS-TE Tunnel Data Model Subtree Structure

This document defines the YANG data model subtree for TE tunnel configuration, state, RPCs and notifications as follows:

```
   +--rw tunnels-cfg!
   |  +--rw tunnel* [name type]
   |     +--rw name                       string
   |     +--rw type                       mtt:tunnel-type
   |     +--rw identifier?                uint16
   |     +--rw description?               string
   |     +--rw admin-mode?                enumeration
   |     +--rw path-protection?           boolean
   |     +--rw backup?                    boolean
   |     +--rw load-share?                uint32
   |     +--rw auto-bandwidth!
   |     |  +--rw overflow-threshold?      uint32
   |     |  +--rw overflow-limit?          uint8
   |     |  +--rw underflow-threshold?     uint32
   |     |  +--rw underflow-limit?         uint8
   |     |  +--rw collect-only?            boolean
   |     |  +--rw application-frequency?   uint32
   |     |  +--rw bandwidth-limit
   |     |     +--rw min-limit?   uint32
   |     |     +--rw max-limit?   uint32
   |     +--rw (forwarding-property)?
   |     |  +--:(forwarding-class)
   |     |  |  +--rw forwarding-class
   |     |  |     +--rw class?   uint8
   |     |  +--:(forwarding-group)
   |     |     +--rw forwarding-group
   |     |        +--rw classes*   uint8
   |     +--rw (routing-proprties)?
   |     |  +--:(autoroute)
   |     |  |  +--rw autoroute!
   |     |  |     +--rw routing-afs*       inet:ip-version
   |     |  |     +--rw (metric-type)?
   |     |  |        +--:(metric)
   |     |  |        |  +--rw metric?            uint32
   |     |  |        +--:(relative-metric)
   |     |  |        |  +--rw relative-metric?   uint32
   |     |  |        +--:(absolute-metric)
   |     |  |           +--rw absolute-metric?   uint32
   |     |  +--:(forwarding-adjacency)
   |     |     +--rw forwarding-adjacency!
   |     |        +--rw holdtime?      uint32
   |     |        +--rw routing-afs*   inet:ip-version
   |     +--rw backup-bandwidth
   |     |  +--rw value?   uint32
   |     |  +--rw type?    mtt:backup-bandwidth-type
   |     +--rw bidirectional
   |     |  +--rw association
   |     |     +--rw id?              uint32
   |     |     +--rw source?          inet:ip-address
   |     |     +--rw global-source?   inet:ip-address
   |     |     +--rw type?            mtt:bidir-association-type
   |     +--rw destination* [address]
   |     |  +--rw address    inet:ip-address
   |     |  +--rw paths* [index]
   |     |     +--rw index             uint8
   |     |     +--rw (type)?
   |     |     |  +--:(dynamic)
   |     |     |  |  +--rw dynamic?          boolean
   |     |     |  +--:(explicit)
   |     |     |     +--rw explicit
   |     |     |        +--rw hops* [index]
   |     |     |           +--rw index           uint8
   |     |     |           +--rw hop-address?   mtt:hop-address-type
   |     |     |           +--rw hop-action?    mtt:hop-action-type
   |     |     +--rw igp-constraint
   |     |     |  +--rw igp?          enumeration
   |     |     |  +--rw area-level?   uint32
   |     |     +--rw no-cspf?          boolean
   |     |     +--rw lockdown?         boolean
   |     +--rw (tunnel-type)?
   |        +--:(p2p-properties)
   |        |  +--rw p2p-properties* [index]
   |        |     +--rw destination?          leafref
   |        |     +--rw index                 leafref
   |        |     +--rw source?               inet:ip-address
   |        |     +--rw fast-reroute!
   |        |     |  +--rw bandwidth-protect?   boolean
   |        |     |  +--rw node-protect?        boolean
   |        |     +--rw record-route?         boolean
   |        |     +--rw signaled-name?        string
   |        |     +--rw priority
   |        |     |  +--rw setup?   uint8
   |        |     |  +--rw hold?    uint8
   |        |     +--rw soft-preemption?      boolean
   |        |     +--rw signaled-bandwidth
   |        |     |  +--rw type?    mtt:bandwidth-type
   |        |     |  +--rw value?   uint32
   |        |     +--rw (style)?
   |        |     |  +--:(affinity-hex)
   |        |     |  |  +--rw affinity-hex
   |        |     |  |     +--rw value?   uint32
   |        |     |  |     +--rw mask?    uint32
   |        |     |  +--:(affinity-name)
   |        |     |     +--rw affinity-name
   |        |     |        +--rw constraints* [action]
   |        |     |           +--rw action   mtt:affinity-action-type
   |        |     |           +--rw constraint
   |        |     |              +--rw affinity-list* [name]
   |        |     |                 +--rw name    string
   |        |     +--rw path-selection
   |        |     |  +--rw cost-limit?   uint32
   |        |     |  +--rw hop-limit?    uint32
   |        |     |  +--rw metric?       mtt:path-metric-type
   |        |     |  +--rw tiebreaker?   mtt:path-tiebreaker-type
   |        |     +--rw bfd!
   |        |     |  +--rw type?                mtt:bfd-type
   |        |     |  +--rw encap-mode?          mtt:bfd-encap-mode-type
   |        |     |  +--rw bringup-timeout?     uint32
   |        |     |  +--rw dampening?           uint32
   |        |     |  +--rw lsp-ping
   |        |     |  |  +--rw disable?    boolean
   |        |     |  |  +--rw interval?   uint32
   |        |     |  +--rw minimum-interval?   uint32
   |        |     |  +--rw multiplier?         uint32
   |        |     +--rw logging-events
   |        |        +--rw (type)?
   |        |           +--:(all)
   |        |           |  +--rw all!
   |        |           +--:(bfd-status)
   |        |           |  +--rw bfd-status!
   |        |           +--:(link-status)
   |        |           |  +--rw link-status!
   |        |           +--:(lsp-status)
   |        |           |  +--rw lsp-status!
   |        |           |     +--rw all?      boolean
   |        |           |     +--rw events*   mtt:lsp-status-event-type
   |        |           +--:(cspf-failure)
   |        |              +--rw cspf-failure!
   |        +--:(p2mp-properties)
   |           +--rw p2mp-properties* [lsp-index]
   |              +--rw lsp-index             uint32
```

```
    |              +--rw p2mp-path-group* [path-index]
    |              |  +--rw path-index     uint32
    |              |  +--rw destination?   leafref
    |              |  +--rw index?         leafref
    |              +--rw source?               inet:ip-address
    |              +--rw fast-reroute!
    |              |  +--rw bandwidth-protect?   boolean
    |              |  +--rw node-protect?        boolean
    |              +--rw record-route?         boolean
    |              +--rw signaled-name?        string
    |              +--rw priority
    |              |  +--rw setup?   uint8
    |              |  +--rw hold?    uint8
    |              +--rw soft-preemption?      boolean
    |              +--rw signaled-bandwidth
    |              |  +--rw type?    mtt:bandwidth-type
    |              |  +--rw value?   uint32
    |              +--rw (style)?
    |              |  +--:(affinity-hex)
    |              |  |  +--rw affinity-hex
    |              |  |     +--rw value?   uint32
    |              |  |     +--rw mask?    uint32
    |              |  +--:(affinity-name)
    |              |     +--rw affinity-name
    |              |        +--rw constraints* [action]
    |              |           +--rw action   mtt:affinity-action-type
    |              |           +--rw constraint
    |              |              +--rw affinity-list* [name]
    |              |                 +--rw name    string
    |              +--rw path-selection
    |              |  +--rw cost-limit?   uint32
    |              |  +--rw hop-limit?    uint32
    |              |  +--rw metric?       mtt:path-metric-type
    |              |  +--rw tiebreaker?   mtt:path-tiebreaker-type
    |              +--rw bfd!
    |              |  +--rw type?                mtt:bfd-type
    |              |  +--rw encap-mode?          mtt:bfd-encap-mode-type
    |              |  +--rw bringup-timeout?     uint32
    |              |  +--rw dampening?           uint32
    |              |  +--rw lsp-ping
    |              |  |  +--rw disable?    boolean
    |              |  |  +--rw interval?   uint32
    |              |  +--rw minimum-interval?   uint32
    |              |  +--rw multiplier?         uint32
    |              +--rw logging-events
    |                 +--rw (type)?
    |                    +--:(all)
    |                    |  +--rw all!
```

```
   |                     +--:(bfd-status)
   |                     |  +--rw bfd-status!
   |                     +--:(link-status)
   |                     |  +--rw link-status!
   |                     +--:(lsp-status)
   |                     |  +--rw lsp-status!
   |                     |     +--rw all?      boolean
   |                     |     +--rw events*   mtt:lsp-status-event-type
   |                     +--:(cspf-failure)
   |                        +--rw cspf-failure!


   +---x tunnels-state
   |  +---x tunnels-state-to-be-defined

   +---x tunnels-rpc
   |  +---x tunnels-rpcs-to-be-defined

   +---n tunnels-notif
   |  +---n tunnels-notifications-to-be-defined
```

7.1. MPLS-TE Tunnel Lists

The data model for tunnels-cfg presented in this document uses a flat list of tunnels. Each tunnel in the list is identified by its name and a mandatory "tunnel-type".

There is one list for configured MPLS-TE tunnels ("/tunnels-cfg/tunnel"), and a separate list for the operational state of all MPLS-TE tunnels ("/tunnels-state/tunnel").

8. MPLS-TE Tunnel LSPs Data Model Subtree Structure

This document defines the YANG data model subtree for TE LSP configuration, state, RPCs and notifications as

the following tree structure:

```
      +--rw lsps-cfg
      |  +--rw lsp* [name]
      |  |  +--rw name               string
      |  |  +--rw lsp-configuration-to-be-defined
      +--ro lsps-state
      |  +--ro lsp* [name]
      |  |  +--ro name               string
      |  |  +--ro lsp-operational-state-to-be-defined

      +---x lsps-rpc
      |  +---x lsp-rpcs-to-be-defined

      +---n lsps-notif
      |  +---n lsp-notifications-to-be-defined
```

8.1. MPLS-TE Tunnel LSP Lists

The data model for lsps-cfg presented in this document uses a flat list of lsps. Each LSP in the list is identified by its name.

There is one list for MPLS-TE tunnel LSP configurations ("/lsps-cfg/lsp"), and a separate list for the operational state of all MPLS-TE tunnel LSPs ("/lsps-state/lsp").

9. MPLS-TE Links Data Model Subtree Structure

This document defines the YANG data model subtree for TE link configuration, state, RPCs and notifications as follows:

```
   +--rw links-cfg
   |  +--rw link* [name]
   |     +--rw name                        string
   |     +--rw admin-weight?               uint32
   |     +--rw reservable-bandwidths
   |     |  +--rw bc-model?                   mtt:bc-model-type
   |     |  +--rw value-type?                 boolean
   |     |  +--rw bandwidth-constraints
   |     |     +--rw maximum-reservable?   uint32
   |     |     +--rw bc0?                  uint32
   |     |     +--rw bc1?                  uint32
   |     |     +--rw bc2?                  uint32
   |     |     +--rw bc3?                  uint32
   |     |     +--rw bc4?                  uint32
   |     |     +--rw bc5?                  uint32
   |     |     +--rw bc6?                  uint32
   |     |     +--rw bc7?                  uint32
   |     +--rw flooding-thresholds
   |     |  +--rw (type)?
   |     |     +--:(single-step)
   |     |     |  +--rw up-step?      uint8
   |     |     |  +--rw down-step?    uint8
   |     |     +--:(multiple-steps)
   |     |        +--rw up-steps* [value]
   |     |        |  +--rw value    uint8
   |     |        +--rw down-steps* [value]
   |     |           +--rw value    uint8
   |     +--rw link-affinities
   |     |  +--rw (type)?
   |     |     +--:(bitmap)
   |     |     |  +--rw bitmap?            uint32
   |     |     +--:(bitmap-extended)
   |     |     |  +--rw bitmap-extended?   uint32
   |     |     +--:(names)
   |     |        +--rw names* [name]
   |     |           +--rw name    string
   |     +--rw shared-risk-link-group* [name]
   |     |  +--rw name    string
   |     +--rw fast-reroute-backup
   |        +--rw (type)?
   |           +--:(configured-backups)
   |           |  +--rw configured-backups* [name]
   |           |     +--rw name    string
   |           +--:(auto-backup)
   |              +--rw name?                     string
   |              +--rw source?                   inet:ip-address
   |              +--rw fast-reroute!
   |              |  +--rw bandwidth-protect?   boolean
   |              |  +--rw node-protect?        boolean
   |              +--rw record-route?             boolean
   |              +--rw signaled-name?            string
   |              +--rw priority
   |              |  +--rw setup?   uint8
   |              |  +--rw hold?    uint8
   |              +--rw soft-preemption?          boolean
   |              +--rw signaled-bandwidth
   |              |  +--rw type?    mtt:bandwidth-type
   |              |  +--rw value?   uint32
   |              +--rw (style)?
   |              |  +--:(affinity-hex)
   |              |  |  +--rw affinity-hex
   |              |  |     +--rw value?   uint32
   |              |  |     +--rw mask?    uint32
   |              |  +--:(affinity-name)
   |              |     +--rw affinity-name
   |              |        +--rw constraints* [action]
   |              |           +--rw action    mtt:affinity-action-type
   |              |           +--rw constraint
   |              |              +--rw affinity-list* [name]
   |              |                 +--rw name    string
   |              +--rw logging-events
   |              |  +--rw (type)?
   |              |     +--:(all)
   |              |     |  +--rw all!
   |              |     +--:(bfd-status)
   |              |     |  +--rw bfd-status!
   |              |     +--:(link-status)
   |              |     |  +--rw link-status!
   |              |     +--:(lsp-status)
   |              |     |  +--rw lsp-status!
   |              |     |     +--rw all?      boolean
   |              |     |     +--rw events*   mtt:lsp-status-event-type
   |              |     +--:(cspf-failure)
   |              |        +--rw cspf-failure!
   |              +--rw auto-bandwidth!
   |              |  +--rw overflow-threshold?      uint32
   |              |  +--rw overflow-limit?          uint8
   |              |  +--rw underflow-threshold?     uint32
   |              |  +--rw underflow-limit?         uint8
   |              |  +--rw collect-only?            boolean
   |              |  +--rw application-frequency?   uint32
   |              |  +--rw bandwidth-limit
   |              |     +--rw min-limit?   uint32
   |              |     +--rw max-limit?   uint32
   |              +--rw (routing-proprties)?
   |              |  +--:(autoroute)
   |              |  |  +--rw autoroute!
   |              |  |     +--rw routing-afs*       inet:ip-version
   |              |  |     +--rw (metric-type)?
   |              |  |        +--:(metric)
   |              |  |        |  +--rw metric?            uint32
   |              |  |        +--:(relative-metric)
   |              |  |        |  +--rw relative-metric?   uint32
   |              |  |        +--:(absolute-metric)
   |              |  |           +--rw absolute-metric?   uint32
   |              |  +--:(forwarding-adjacency)
   |              |     +--rw forwarding-adjacency!
   |              |        +--rw holdtime?      uint32
   |              |        +--rw routing-afs*   inet:ip-version
   |              +--rw (forwarding-property)?
   |              |  +--:(forwarding-class)
   |              |  |  +--rw forwarding-class
   |              |  |     +--rw class?   uint8
   |              |  +--:(forwarding-group)
   |              |     +--rw forwarding-group
   |              |        +--rw classes*   uint8
   |              +--rw link-protection-only?      boolean
   |              +--rw bandwidth-protection?      boolean
   |              +--rw backup-path-computation?
                auto-backup-path-computation-type

   +--ro links-state
   |  +--ro link* [name]
   |  |  +--ro name               string
   |  |  +--ro link-operational-state-to-be-defined

   +---x links-rpc
   |  +---x link-rpcs-to-be-defined

   +---n links-notif
   |  +---n link-notifications-to-be-defined
```

9.1. MPLS-TE Link Lists

The data model for links-cfg presented in this document uses a flat list of links. Each link in the list is identified by its name.

There is one list for MPLS-TE link configurations ("/links-cfg/link"), and a separate list for the operational state of all MPLS-TE links ("/links-state/link").

10. MPLS-TE YANG Generic Types Module

```
module ietf-mpls-te-types {

    namespace "urn:cisco:params:xml:ns:yang:mpls-te-types";

    /* Replace with IANA when assigned */
    prefix "mpls-te-types";

    import ietf-inet-types { prefix inet; }

    organization
      "Cisco Systems
       170 West Tasman Drive
       San Jose, CA 95134-1706
       USA";

    contact
```

```
      "Rakesh Gandhi rgandhi@cisco.com
       Tarek Saad tsaad@cisco.com
       Robert Sawaya rsawaya@cisco.com";

    description
        "This module contains a collection of generally
         useful MPLS-TE specific
         derived YANG data types.";

    revision 2014-10-27 {
        description
            "Initial revision.";
    }

    /* Typedefs for MPLS-TE */

    typedef bc-model-type {
        description
           "Diff-Serve TE Bandwidth Constraint model type.";
        type enumeration {
            enum rdm {
                description
                "Russian Doll bandwidth constraint model type.";
            }
            enum mam {
                description
                "Maximum Allocation bandwidth constraint model type.";
            }
            enum mar {
                description
                "Maximum Allocation with Reservation
                bandwidth constraint model type.";
            }
        }
        default rdm;
    }

    typedef bandwidth-type {
        description "MPLS-TE tunnel bandwidth type";
        type enumeration {
            enum CT0;
            enum CT1;
            enum CT2;
            enum CT3;
            enum CT4;
            enum CT5;
            enum CT6;
            enum CT7;
        }
        default CT0;
    }

    typedef lsp-status-event-type {
        description "Tunnel LSP status event type.";
        type enumeration {
            enum bandwidth-change;
            enum insufficient-bandwidth;
            enum record-route;
            enum reroute;
            enum state;
            enum switchover;
        }
    }

    typedef bandwidth-unit-type {
        description "Bandwidth unit type.";
        type enumeration {
            enum Gbps;
            enum Mbps;
            enum Kbps;
        }
        default Kbps;
    }

    typedef backup-bandwidth-type {
        description "FRR backup tunnel bandwidth protection type.";
        type enumeration {
            enum BC0;
            enum BC1;
            enum BC2;
            enum BC3;
            enum BC4;
            enum BC5;
            enum BC6;
            enum BC7;
            enum BC-any;
        }
        default BC-any;
    }

    typedef tunnel-type {
        type enumeration {
            enum p2p {
                description
                    "MPLS-TE point-to-point tunnel type.";
            }
            enum p2mp {
                description
                    "MPLS-TE point-to-multipoint tunnel type.";
            }
        }
        default p2p;
        description
            "Possible MPLS-TE tunnel types, default is point-to-point.";
    }

    typedef hop-address-type {
        type inet:ip-address;
    }

    typedef hop-action-type {
        type enumeration {
            enum include-strict {
                description "Include strict hop.";
            }
            enum include-loose {
                description "Include loose hop.";
            }
            enum exclude {
                description "Exclude strict hop.";
            }
        }
        default "include-strict";
    }

    typedef bfd-type {
        type enumeration {
            enum classical {
                description "BFD classical session type.";
            }
            enum seamless {
                description "BFD seamless session type.";
            }
        }
        default "classical";
    }

    typedef path-metric-type {
        type enumeration {
            enum igp;
            enum te;
        }
        default igp;
        description "Path metric for CSPF.";
    }

    typedef path-tiebreaker-type {
        type enumeration {
            enum min-fill;
            enum max-fill;
            enum random;
        }
        default min-fill;
        description
            "Possible CSPF path tiebreakers for MPLS-TE tunnels.";
    }

    typedef bidir-association-type {
        type enumeration {
            enum corouted;
            enum non-corouted;
        }
        default non-corouted;
        description
            "Possible types of bidirectional tunnel association.";
    }

    typedef bfd-encap-mode-type {
        type enumeration {
            enum gal;
            enum ip;
        }
        default ip;
        description
            "Possible BFD transport modes when running over MPLS-TE
             LSPs.";
    }

        typedef affinity-action-type {
            type enumeration {
                enum include;
                enum exclude;
                enum include-strict;
                enum exclude-all;
            }
            description
                "Possible handling for affinity.";
        }
    }
```

11. MPLS TE Yang Module

```
    module ietf-mpls-te {

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

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

    import ietf-inet-types { prefix inet; }
    import ietf-mpls-te-types { prefix mtt; }

    organization
      "Cisco Systems
       170 West Tasman Drive
       San Jose, CA 95134-1706
       USA";

    contact
      "Rakesh Gandhi rgandhi@cisco.com
       Tarek Saad tsaad@cisco.com
       Robert Sawaya rsawaya@cisco.com";

    description
        "Data model for MPLS-TE configuration, state, execution and
         notifications.";

    revision 2014-10-27 {
        description
            "Initial revision.";
    }

    /* Groupings for MPLS-TE */
    grouping lsp-properties {

        choice tunnel-type {
            /* Point-to-point LSP properties */
            list p2p-properties {
                when "/tunnels-cfg/tunnel/type = p2p";
                key "index";
                uses path-option-reference;
                    description "An index identifying a set of LSP
                                 properties";
                uses signaling-properties;
                uses affinity-properties;
                uses path-computation-properties;
                uses bfd-properties;
                uses logging-events;
            }

            /* Point-to-multipoint LSP properties */
            list p2mp-properties {
                when "/tunnels-cfg/tunnel/type = p2mp";
                key "lsp-index";
                leaf lsp-index {
                    description "An index identifying a set of LSP
                                 properties";
                    type uint32;
                }
                list p2mp-path-group {
                    key "path-index";
                    description "Index pointing to the configured
                                path as specified in
                                the path-option-reference";
                    leaf path-index {
                        type uint32;
                    }
                    /* definition of the destination
                      and associated path  */
                    uses path-option-reference;
                }
                uses signaling-properties;
                uses affinity-properties;
                uses path-computation-properties;
                uses bfd-properties;
                uses logging-events;
            }
        }
    }

    grouping path-option-reference {
        leaf destination {
            description "A reference to an MPLS-TE tunnel destination";
            type leafref {
                path "/tunnels-cfg/tunnel/destination/address";
            }
        }
        leaf index {
            description "A reference to an MPLS-TE tunnel path-option";
            type leafref {
                path "/tunnels-cfg/tunnel/destination[address=current()
                /../destination]/paths/index";
            }
        }
    }
    grouping path-properties {
        choice type {
            leaf dynamic {
                description
                    "A CSPF dynamically computed path";
                type boolean;
            }
            container explicit {
                description
                    "An operator specified explicit path";
                list hops {
                    key "index";
                    leaf index {
                        type uint8;
                    }
                    leaf hop-address {
                        description
                            "An IP hop address";
                        type mtt:hop-address-type;
                    }
                    leaf hop-action {
                        description
                            "An IP hop action.";
                        type mtt:hop-action-type;
                    }
                }
            }
        }
        container igp-constraint {
            leaf igp {
                description
                    "Constrains the computed path to a specific IGP.";
                type enumeration {
                    enum ospf;
                    enum isis;
                }
            }
            leaf area-level {
                description
                    "Constrains the computed path to a specific IGP
                     area or level";
                type uint32;
            }
        }
        leaf no-cspf {
            description
                "Indicates no validation checks to be attempted on this
                 path";
            type boolean;
        }
        leaf lockdown {
            description
                "Indicates no reoptimization to be attempted for this
                 path.";
            type boolean;
        }
    }

    grouping bfd-properties {
        container bfd {
            presence "Enables BFD fast-detect on the tunnel.";
            leaf type {
                type mtt:bfd-type;
            }
            leaf encap-mode {
                type mtt:bfd-encap-mode-type;
            }
            leaf bringup-timeout {
                type uint32;
            }
            leaf dampening {
                type uint32;
            }
            container lsp-ping {
                leaf disable {
                    type boolean;
                    default false;
                }
                leaf interval {
                    type uint32;
                }
            }
            leaf minimum-interval {
                type uint32;
            }
            leaf multiplier {
                type uint32;
            }
        }
    }

    grouping path-computation-properties {
        container path-selection {
            leaf cost-limit {
                description
                    "The MPLS-TE tunnel path cost limit.";
                type uint32;
            }
            leaf hop-limit {
                description
                    "The MPLS-TE tunnel path hop limit.";
                type uint32;
            }
            leaf metric {
                description
                    "The MPLS-TE tunnel path metric type.";
                type mtt:path-metric-type;
            }
            leaf tiebreaker {
                type mtt:path-tiebreaker-type;
            }
        }
    }

    grouping signaling-properties {
        description "LSP signaling properties.";
        leaf source {
            description
                "LSP source address.";
            type inet:ip-address;
        }
        container fast-reroute {
            presence "Requests FRR local protection on LSRs";
            leaf bandwidth-protect {
                description
                    "Requests FRR bandwidth protection on LSRs";
                type boolean;
            }
            leaf node-protect {
                description
                    "Requests FRR node protection on LSRs";
                type boolean;
            }
        }
        leaf record-route {
            description
                "Requests path RRO recording in RSVP PATH message.";
            type boolean;
        }
        leaf signaled-name {
            description
                "Sets the session name to use in the session attribute
                 object.";
            type string;
        }
        container priority {
            description
                "Sets the setup/hold priority to use in the session
                 attribute object";
            leaf setup {
                type uint8;
            }
            leaf hold {
                type uint8;
            }
        }
        leaf soft-preemption {
            description
                "Requests soft-preemption in session
                 attributes object at
                 at traversed LSR(s).";
            type boolean;
        }
        container signaled-bandwidth {
            description
                "Sets the requested bandwidth";
            leaf type {
                type mtt:bandwidth-type;
            }
            leaf value {
                type uint32;
            }
        }
    }

    grouping logging-events {
        container logging-events {
            choice type {
                container all {
                    presence "Enable all MPLS-TE tunnel event logging.";
                }
                container bfd-status {
                    presence "Enable MPLS-TE tunnel BFD specific event
                              logging.";
                }
                container link-status {
                    presence "Enable link status event logging.";
                }
                container lsp-status {
                    presence "Enable LSP status event logging.";
                    leaf all {
                        type boolean;
                    }
                    leaf-list events {
                        type mtt:lsp-status-event-type;
                    }
                }
                container cspf-failure {
                    presence "Enable MPLS-TE tunnel CSPF failure event
                              logging.";
                }
            }
        }
    }

    grouping affinity-properties {
        choice style {
            container affinity-hex {
                leaf value {
                    type uint32;
                }
                leaf mask {
                    type uint32;
                }
            }
            container affinity-name {
                list constraints {
                    key "action";
                    leaf action {
                        type mtt:affinity-action-type;
                    }
                    container constraint {
                        list affinity-list {
                            key "name";
                            leaf name {
                                type string;
                            }
                        }
                    }
                }
            }
        }
    }

    grouping routing-proprties {
        choice routing-proprties {
            description
                "Announces the tunnel to IGP as either autoroute or
                 forwarding adjacency.";
            container autoroute {
                presence "Enables autoroute announce.";
                description
                    "Announce the MPLS-TE tunnel as autoroute to IGP for
                     use as IGP shortcut";
                leaf-list routing-afs {
                     type inet:ip-version;
                }
                choice metric-type {
                    leaf metric {
                        type uint32;
                    }
                    leaf relative-metric {
                        type uint32;
                    }
                    leaf absolute-metric {
                        type uint32;
                    }
                }
            }
            container forwarding-adjacency {
                presence "Enables forwarding adjacency on the tunnel.";
                description
                    "Announce the MPLS-TE tunnel as
                     forwarding adjacency.";
                leaf holdtime {
                    description
                        "Holdtime in seconds after tunnel becomes UP.";
                    type uint32;
                }
                leaf-list routing-afs {
                     type inet:ip-version;
                }
            }
        }
    }

    grouping auto-bandwidth {
        container auto-bandwidth {
            presence "Enabled MPLS-TE tunnel auto-bandwidth feature";
            description
                "MPLS-TE tunnel auto-bandwidth configuration
                parameters.";
            leaf overflow-threshold {
                description
                    "Auto-bandwidth change percent to trigger overflow";
                type uint32;
            }
            leaf overflow-limit {
                description
                    "Auto-bandwidth consecutive collections to trigger
                      overflow";
                type uint8;
            }
            leaf underflow-threshold {
                description
                    "Auto-bandwidth change percent to
                     trigger underflow";
                type uint32;
            }
            leaf underflow-limit {
                description
                    "Auto-bandwidth consecutive collections to trigger
                      underflow";
                type uint8;
            }
            leaf collect-only {
                description
                    "Auto-bandwidth collection only mode";
                type boolean;
            }
            leaf application-frequency {
                description
                    "Auto-bandwidth application interval in seconds";
                type uint32;
            }
            container bandwidth-limit {
                leaf min-limit {
                    type uint32;
                }
                leaf max-limit {
                    type uint32;
                }
            }
        }
    }

    grouping backup-bandwidth {
        container backup-bandwidth {
            when "/tunnels-cfg/tunnel/backup = true";
            description
                "The backup bandwidth that can be protected by this
                 backup tunnel.";
            leaf value {
                type uint32;
            }
            leaf type {
                type mtt:backup-bandwidth-type;
            }
        }
    }

    grouping forwarding-properties {
        description "Policy for using tunnel in forwarding.";
        choice forwarding-property {
            container forwarding-class {
                leaf class {
                    type uint8;
                    default 0;
                }
            }
            container forwarding-group {
                leaf-list classes {
                    type uint8;
                }
            }
        }
    }

    grouping flooding-thresholds {
        description "Flooding threshold values.";
        container flooding-thresholds {
            choice type {
                case single-step {
                    leaf up-step { type uint8; }
                    leaf down-step { type uint8; }
                }
                case multiple-steps {
                    list up-steps {
                        key "value";
                        description "Percentage bandwidth
                            exceeded that causes flooding";
                        leaf value { type uint8; }
                    }
                    list down-steps {
                        key "value";
                        description "Percentage bandwidth
                               crossed that causes flooding";
                            leaf value { type uint8; }
                    }
                }
            }
        }
    }
    /* MPLS-TE Tunnel Template Configuration Data */
    grouping tunnel-template {
        leaf name {
            description "MPLS-TE tunnel-template name.";
            type string;
        }
        uses signaling-properties;
        uses affinity-properties;
        uses logging-events;
        uses auto-bandwidth;
        uses routing-proprties;
        uses forwarding-properties;
    }

    /* MPLS-TE Tunnel Configuration Data */
    container tunnels-cfg {
        description
            "Configuration, operational, notification,
              and RPC data model
             for MPLS-TE tunnels.";
        presence "Enables MPLS-TE Global mode.";
        list tunnel {
            key "name type";
            unique "identifier";
            description "MPLS-TE tunnel.";
            leaf name {
                type string;
                description "MPLS-TE tunnel name.";
            }
            leaf type {
                description "MPLS-TE tunnel type.";
                type mtt:tunnel-type;
            }
            leaf identifier {
                description
                    "MPLS-TE tunnel Identifier.";
                type uint16;
            }
            leaf description {
                description
                    "MPLS-TE tunnel description.";
                type string;
            }
            leaf admin-mode {
                description "MPLS-TE tunnel administrative state";
                type enumeration {
                    enum up;
                    enum down;
                }
                default up;
            }
            leaf path-protection {
                description "Enable MPLS-TE tunnel
                     end-to-end path-protection.";
                type boolean;
            }
            leaf backup {
                description "Tunnel is being used
                       as fast reroute backup.";
                type boolean;
                default false;
            }
            leaf load-share {
                description "ECMP tunnel forwarding load-share factor.";
                type uint32;
            }
            uses auto-bandwidth;
            uses forwarding-properties;
            uses routing-proprties;
            uses backup-bandwidth;
            container bidirectional {
                description
                    "MPLS-TE associated
                             bidirectional tunnel attributes.";
                container association {
                    leaf id {
                        description
                            "The MPLS-TE tunnel
                              association identifier.";
                        type uint32;
                    }
                    leaf source {
                        description
                            "The MPLS-TE tunnel association source.";
                        type inet:ip-address;
                    }
                    leaf global-source {
                        description
                            "The MPLS-TE tunnel association global
                             source.";
                        type inet:ip-address;
                    }
                    leaf type {
                        description "The MPLS-TE
                               tunnel association type.";
                        type mtt:bidir-association-type;
                    }
                }
            }

            /* List of destinations and path(s) */
            list destination {
                key "address";
                description
                    "The MPLS-TE tunnel destination address.";
                leaf address {
                    type inet:ip-address;
                }
                list paths {
                    key "index";
                    leaf index {
                        type uint8;
                    }
                    uses path-properties;
                }
            }

            uses lsp-properties;
        }
    }

    /* MPLS-TE Global Configuration Data */
    container global-cfg {

        description
            "Configuration data model for Global System-wide MPLS
             Traffic Engineering.";
        presence "Enables MPLS-TE Global mode.";

        grouping load-share-properties {
            container load-share-properties {
                description "ECMP tunnel forwarding
                            load-share properties.";
                container unequal {
                    presence "ECMP tunnels to
                              load-share forwarding of
                              traffic unequally.";

                    leaf bandwidth-based {
                         type boolean;
                         description
                             "ECMP tunnels to load-share
                             forwarding of traffic
                             based on tunnel bandwidth.";
                    }
                }
            }
        }

        grouping global-timers {
            container global-timers {
                description
                    "Global system-wide TE timer values.";
                leaf lsp-bw-hold-delay {
                     type  uint32 ;
                     description
                         "Bandwidth hold interval for
                         LSP admission in seconds.";
                }
                leaf lsp-preemption-delay {
                     type  uint32 ;
                     description
                         "Delay in seconds to preempt
                          LSPs after preemption
                         event triggered.";
                }
                leaf topology-holddown-sigerr {
                     type  uint32 ;
                     description
                         "Link holddown in topology database for CSPF.
                         Delay in seconds after signaling error.";
                }
            }
        }

        grouping global-reoptimization {
            description
                "Global periodic LSP reoptimization parameters.";
            container global-reoptimization {
                presence "Enable TE tunnel reoptimization globally
                         on the node.";

                leaf interval {
                     type  uint32;
                     description
                         "Periodic reoptimization
                          interval in seconds.";
                }
                leaf installation-delay {
                     type  uint32;
                     description
                         "Delay in seconds before
                         installing reoptimzing LSP
                        in forwarding to carry traffic.";
                }
                leaf cleanup-delay {
                     type  uint32;
                     description
                         "Delay in seconds before
                          removing reoptimized LSP
                         in forwarding.";
                }
            }
        }

        grouping topology-flooding {
            description
                "Global periodic TE topology flooding parameters.";
            container topology-flooding {
                presence "Enable TE topology
                flooding globally on the node.";

                leaf flooding-interval {
                     type  uint32 ;
                     description
                         "Periodic topology flooding
                         interval in seconds.";
                }

                uses flooding-thresholds;
            }
        }

        grouping affinity-map {
            description
                "Mapping of affinity name and value.";
            leaf name {
                description
                    "Name of the affinity.";
                type string;
            }
            choice type  {
                leaf value {
                    type uint32;
                }
                /* TBA: 256 bit value */
                leaf bit-index {
                    type uint32;
                }
            }
        }

        grouping affinity-maps {
            description
                "Mapping of Affinity name and value.";

            container affinity-maps {
                list affinity-maps {
                    description "MPLS-TE affinity-maps.";
                    key "name";
                    uses affinity-map;
                }
            }
        }

        grouping srlg-map {
            description
                "Mapping of SRLG name, value and admin-weight.";
            leaf name {
                type string;
            }
            leaf value {
                type uint32;
            }
            leaf admin-weight {
                type uint32;
            }
        }

        grouping srlg-maps {
            description
                "Mapping of SRLG name, value and admin-weight.";
            container srlg-maps {
                list srlg-maps {
                    description "MPLS-TE srlg-map.";
                    key "name";
                    uses srlg-map;
                }
            }
        }

        grouping softpreemption-properties {
            description
                "Softpreemption properties.";
            container softpreemption-te {
                presence "Enable soft-preemption on the node.";

                leaf softpremption-timeout {
                     type  uint32 ;
                     description
                         "Delay in seconds to teardown soft-preempted
                         LSPs after soft-preemption event.";
                }
            }
        }

        grouping diff-serve-te-properties {
            description
                "Diff-Serve TE properties.";
            container ds-te {
                presence "Enable Diff-Serve TE on the node.";

                leaf bc-model {
                     description
                         "Diff-Serve TE bandwidth
                          constraint model type.";
                     type mtt:bc-model-type;
                }
                list te-class {
                    description
                        "Diff-Serve TE TE-class mapping.";
                    key "class-index";
                    leaf class-index {
                         description
                            "TE-class index.";
                         type uint8;
                    }
                    leaf priority {
                         description
                             "LSP setup or hold priority.";
                         type uint8;
                    }
                    leaf bc-value {
                         description
                             "Bandwidth Constraint pool value.";
                         type uint8;
                    }
                }
            }
        }

        grouping auto-bandwidth-properties {
            description
                "Auto-bandwidth adjustment properties.";
            container auto-bw {
                presence "Enable auto-bandwidth on the node.";
```

```
                leaf stats-collection-interval {
                    description
                        "Auto-bandwidth statistics collection
                        interval in seconds.";
                    type uint32;
                }
            }
        }

        grouping auto-mesh-properties {
            description
                "Auto-tunnel mesh-group properties.";
            leaf mesh-group {
                 description
                    "Value of the mesh-group.";
                 type uint32;
            }
            leaf template-name {
                 description
                    "Name of the template to use
                     for tunnel properties.";
                 type string;
            }
            leaf one-hop-only {
                 description
                    "Limit tunnel to one-hop away nodes only.";
                 type boolean;
            }
        }

        grouping auto-backup-properties {
            description
                "Auto-tunnel backup properties.";
            leaf template-name {
                 description
                    "Name of the template to use for
                     tunnel properties.";
                 type string;
            }
            leaf nhop-only {
                 description
                    "Limit tunnel to link protection only.";
                 type boolean;
            }
        }

        grouping tail-signaling {
            description
```

```
                "Signaling properties for LSP tail-end.";
            container tail-signaling {
                leaf advertise-explicit-null {
                     type boolean;
                }
            }
        }

        uses path-computation-properties;
        uses bfd-properties;
        uses logging-events;

        /* template-type is a leaf. mtt:tunnel-type is also leaf.
         */
        list tunnel-templates {
            description "MPLS-TE templates.";
            key "name";
            uses tunnel-template;
        }

        list auto-tunnel-mesh {
            description "MPLS-TE auto-tunnel mesh-groups.";
            key "mesh-group";
            uses auto-mesh-properties;
        }

        uses auto-backup-properties;
        uses load-share-properties;
        uses global-timers;
        uses topology-flooding;
        uses global-reoptimization;
        uses softpreemption-properties;
        uses affinity-maps;
        uses srlg-maps;
        uses diff-serve-te-properties;
        uses auto-bandwidth-properties;
        uses tail-signaling;
    }

    /* MPLS-TE Link Configuration Data */
    container links-cfg {
        description
            "Configuration data model for MPLS-TE links.";

        typedef  auto-backup-path-computation-type {
            type enumeration {
                enum srlg-none;
                enum srlg-strict;
                enum srlg-preferred;
                enum srlg-weighted;
            }
        }

        grouping bandwidth-constraints {
            container bandwidth-constraints {
                leaf maximum-reservable {type uint32;}
                leaf bc0 {type uint32;}
                leaf bc1 {type uint32;}
                leaf bc2 {type uint32;}
                leaf bc3 {type uint32;}
                leaf bc4 {type uint32;}
                leaf bc5 {type uint32;}
                leaf bc6 {type uint32;}
                leaf bc7 {type uint32;}
            }
        }

        list link {
            key "name";
            description "MPLS-TE links.";
            leaf name {
                type string;
                description "MPLS-TE link name.";
            }

            leaf admin-weight {
                description "MPLS-TE admin-weight.";
                type uint32;
            }

            container reservable-bandwidths {
                description "Reservabale bandwidth values.";
                leaf bc-model {
                     description
                         "Diff-Serve TE bandwidth
                         constraint model type.";
                     type mtt:bc-model-type;
                }
                leaf value-type {
                     /* absolute-value */
                     /* percentage-value */
                     type boolean;
                }
                uses bandwidth-constraints;
            }
            uses flooding-thresholds;

            container link-affinities {
                choice type {
                    case bitmap {
                        leaf bitmap {
                             type uint32;
                        }
                    }
                    case bitmap-extended {
                        /* TBA: 256 bit */
                        leaf bitmap-extended {
                             type uint32;
                        }
                    }
                    case names {
                        list names {
                            key "name";
                            description "List of affinity names.";
                            leaf name {
                                type string;
                            }
                        }
                    }
                }
            }

            list shared-risk-link-group {
                key "name";
                description "List of SRLGs that this link is part of";
                leaf name {
                     type string;
                }
            }

            container fast-reroute-backup {
                choice type {
                    case configured-backups {
                        list configured-backups {
                            key "name";
                            description "List of backup
                            tunnels to protect this link";
                            leaf name {type string;}
                        }
                    }
                    case auto-backup {
                        uses tunnel-template;
                        leaf link-protection-only {
                             type boolean;
                        }
                        leaf bandwidth-protection {
                             type boolean;
                        }
                        leaf backup-path-computation {
                            type auto-backup-path-computation-type;
                        }
                    }
                }
            }
        }
    }


    /* MPLS-TE Tunnel Operational Data */
    container tunnels-oper {
        config "false";
        /* mandatory "true"; */
        description "MPLS-TE tunnel operational state data.";
        list tunnel {
            description "MPLS-TE tunnel.";
            key "name";
            leaf name {
                type string;
                description "MPLS-TE tunnel name.";
            }
        }
    }

    /* MPLS-TE Global Operational Data */
    container global-oper {
        config "false";
    }

    /* MPLS-TE Global Operational Data */
    container links-oper {
        description
            "Operational data model for MPLS-TE links.";
        config "false";
    }

    /* MPLS-TE Tunnel RPCs/execution Data */
    rpc tunnels-rpc {
        description
            "foo bar";
        input {
            leaf input-command {
                type string;
                default "";
                description
                    "The string with which the tape shall
                    be initialized. The
                    leftmost symbol will be at tape
                    coordinate 0.";
            }
        }
        output {
            leaf out-message {
                type string;
                default "";
                description "foo bar";
            }
        }
    }

    /* MPLS-TE Global RPCs/execution Data */
    rpc global-rpc {
        description
            "Execution data for MPLS-TE global.";
    }

    /* MPLS-TE links RPCs/execution Data */
    rpc links-rpc {
        description
            "Execution data for MPLS-TE links.";
    }

    /* MPLS-TE Tunnel Notification Data */
    notification tunnels-notif {
        description
            "The Turing Machine has halted. This means that there is no
            transition rule for the current state and tape symbol.";
        leaf state {
            type mtt:tunnel-type;
            mandatory "true";
        }
    }

    /* MPLS-TE Global Notification Data */
    notification global-notif {
    }

    /* MPLS-TE Links Notification Data */
    notification links-notif {
    }
```

```
}
```

12. IANA Considerations

This document registers a URI 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-te
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-te
namespace:  urn:ietf:params:xml:ns:yang:ietf-mpls-te
prefix:     mpls-te
reference:  RFC XXXX
```

13. 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. Following are the subtrees and data nodes and their sensitivity/vulnerability:

/global-cfg: This module specifies the global MPLS-TE configurations on a device. Unauthorized access to this list could cause the device to ignore packets it should receive and process.

/tunnels-cfg/tunnel: This list specifies the configured MPLS-TE tunnes on a device. Unauthorized access to this list could cause the device to ignore packets it should receive and process.

/lsps-cfg/lsp: This list specifies the configured MPLS-TE LSPs on a device. Unauthorized access to this list could cause the device to ignore packets it should receive and process.

/links-cfg/link: This list specifies the configured MPLS-TE links on a device. Unauthorized access to this list could cause the device to ignore packets it should receive and process.

14. Acknowledgement

TBA.

15. References

15.1. Normative References

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

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

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

[RFC6991] Schoenwaelder, J., "Common YANG Data Types", RFC 6991, July 2013.

15.2. Informative References

[RFC2205] Braden, B., Zhang, L., Berson, S., Herzog, S., and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, September 1997.

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, December 2001.

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A.

Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, June 2011.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, March 2012.

[I-D.ietf-netmod-routing-cfg] Lhotka, L., "A YANG Data Model for Routing Management", draft-ietf-netmod-routing-cfg-16 (work in progress), May 2014.

[I-D.ietf-netmod-rfc6087bis] Bierman, A., "Guidelines for Authors and Reviewers of YANG Data Model Documents", draft-ietf-netmod-rfc6087bis-01 (work in progress), June 2014.

16. Authors' Addresses

Rakesh Gandhi
Cisco Systems, Inc.

Email: rgandhi@cisco.com

Tarek Saad
Cisco Systems, Inc.

Email: tsaad@cisco.com

Robert Sawaya
Cisco Systems, Inc.

Email: rsawaya@cisco.com