Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 30, 2015

X. Chen
Z. Li
S. Hares
Huawei Technologies
R. White
J. Tantsura
Ericsson
October 27, 2014

I2RS Information Model for MPLS TE
draft-chen-i2rs-mpls-te-info-model-00

Abstract

Traditionally MPLS TE networks may be managed via CLI, SNMP or NETCONF. There are two types of TE LSP: static CR-LSP and dynamic TE LSP created by protocol of RSVP-TE. Static CR-LSP is configured with forwarding items such as interface, label and bandwidth, etc. node by node. Dynamic TE LSP is configured with MPLS TE parameters which are used to calculate path and set up TE LSP by protocol. Both configurations are complex.

The Interface to the Routing System's (I2RS) Programmatic interface (draft-ietf-i2rs-architecture) provides an alternate way to control the configuration and diagnose the operation of MPLS TE. These interactions to control MPLS TE links and diagnose their operation include: MPLS TE configuration, MPLS TE protection, traffic switching-over, traffic detection, and fault detection.

This document specifies an information model for MPLS TE to facilitate the definition of a standardized data model which can be used to define interfaces to the MPLS TE from an entity that may even be external to the routing system. Based on standardized data model and interfaces, use cases of MPLS TE defined by draft-huang-i2rs-mpls-te-usecases can be supported.

Requirements Language

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

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Copyright Notice

Table of Contents

1. Introduction

Traditionally MPLS TE networks may be managed via CLI, SNMP or NETCONF. There are two types of TE LSP: static CR-LSP and dynamic TE LSP created by protocol of RSVP-TE. Static CR-LSP is configured with

forwarding items such as interface, label and bandwidth, etc. node by node. Dynamic TE LSP is configured with MPLS TE parameters which are used to calculate path and set up TE LSP by protocol. Both configurations are complex.

With the expansion and complication of modern networks, the necessity for rapid and dynamic control has been increased. The Interface to the Routing System's (I2RS) Programmatic interface ([I-D.ietf-i2rs-architecture]) provides an alternate way to control the configuration and diagnose the operation of MPLS TE and achieve this goal. These interactions to control MPLS TE links and diagnose their operation include: MPLS TE configuration, MPLS TE protection, traffic switching-over, traffic detection, and fault detection.

This document specifies an information model for MPLS TE to facilitate the definition of a standardized data model, which can be used to define interfaces to the MPLS TE from an entity that may even be external to the routing system. Based on standardized data model and interfaces, use cases and requirements for an interface to MPLS TE defined by [I-D.huang-i2rs-mpls-te-usecases] and [I-D.hares-i2rs-usecase-reqs-summary] can be supported.

Please note I2RS utilizes ephemeral configuration plus status information. This draft proposes needs of this ephemeral configuration, and the authors of this draft intend to collaborate with related work on yang configuration for MPLS TE.

2. MPLS TE Data

This section describes the data involved in the MPLS TE information model in detail. MPLS TE data includes information related to Static CR LSPs, dynamic RSVP-TE LSPs and explicit path constraints for setting up RSVP-TE LSPs.

A high-level architecture of the MPLS TE contents is shown as below.

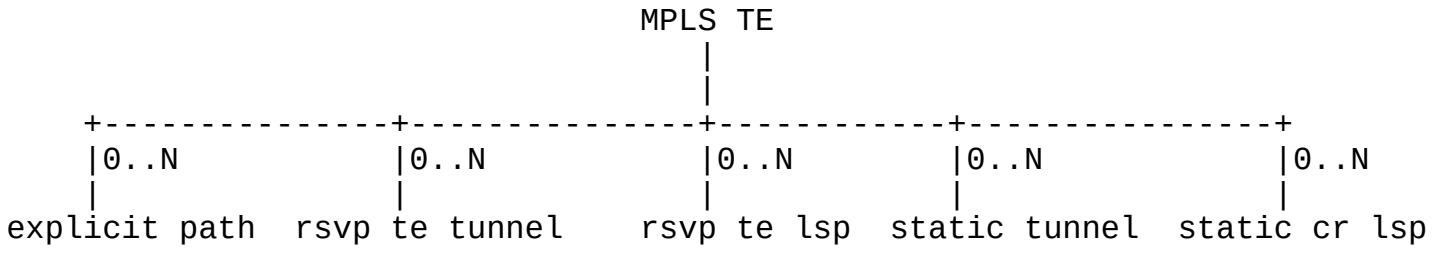

Figure 1: Architecture of MPLS TE information model

2.1. MPLS TE

MPLS TE information model includes information related to Static CR LSPs, dynamic RSVP-TE LSPs and explicit path constraints for setting up RSVP-TE LSPs.

The corresponding YANG description of the top level is below.

```
module: i2rs-mplste
   +--rw mplsTe
      +--rw explicitPaths
      |     ...
      +--rw rsvpTeTunnels
      |     ...
      +--rw rsvpTeLsps
      |     ...
      +--rw staticTunnels
      |     ...
      +--rw staticCRLsps
            ...
```

2.1.1. Explicit Path

The I2RS client should be able to manually calculate a re-optimization of the MPLS TE network and send the new constraints including the calculated path to each node via the I2RS agent with an indication to re-signal the TE LSPs with make-before-break method.

This section describes the information model related to explicit path which is shown in the Yang high-level description and interprets the meaning of each element.

```
      +--rw explicitPaths
      |  +--rw explicitPath* [explicitPathName]
      |     +--rw explicitPathName    string
      |     +--rw explicitPathHops
      |        +--rw explicitPathHop* [mplsTunnelHopIndex]
      |           +--rw mplsTunnelHopIndex       uint32
      |           +--rw mplsTunnelHopIpAddr      inet:ipv4-address
      |           +--rw mplsTunnelHopType?       enumeration
      |           +--rw mplsTunnelHopAddrType?   enumeration
```

o explicitPathName: the name of an explicit path.

o mplsTunnelHopIndex: hop index of an explicit path.

o mplsTunnelHopIpAddr: IP address of one hop.

o mplsTunnelHopType: an LSP route selection types based on the local hop which can be strict, loose and excluding. Strict type: Only an LSP route that includes the local hop can be selected. Loose type: An LSP route that includes the local node is selected preferentially. If the local hop does not meet path limits, it will be not included in the selected route. Excluding type: Only an LSP route that does not include the local hop can be selected.

o mplsTunnelHopAddrType: address type.

2.1.2. RSVP-TE Tunnel

When the network uses dynamic RSVP-TE tunnel the I2RS client can configure the explicit path and other constraints of multiple tunnel paths to the I2RS agent. And the RSVP-TE tunnel information includes the state of the tunnel and the protection-related information.

This section describes the information model related to RSVP-TE tunnel which is shown in the Yang high-level description and interprets the meaning of each element.

```
        +--rw rsvpTeTunnels
        |  +--rw rsvpTeTunnel* [tunnelName]
        |     +--rw tunnelName                    string
        |     +--ro mplsTunnelIngressLSRId?       inet:ipv4-address
        |     +--rw mplsTunnelEgressLSRId?        inet:ipv4-address
        |     +--rw mplsTunnelIndex?              uint16
        |     +--rw mplsTunnelBandwidth?          uint32
        |     +--rw mplsTeTunnelSetupPriority?    uint8
        |     +--rw holdPriority?                 uint8
        |     +--rw hotStandbyEnable?             boolean
        |     +--rw hsbRevertiveMode?             enumeration
        |     +--rw hotStandbyWtr?                uint32
        |     +--rw ordinaryEnable?               boolean
        |     +--rw bestEffortEnable?             boolean
        |     +--rw disableCspf?                  boolean
        |     +--rw tunnelPaths
        |     |  +--rw tunnelPath* [pathType]
        |     |     +--rw pathType             enumeration
        |     |     +--rw explicitPathName?    string
        |     |     +--ro includeAll?          string
        |     |     +--rw includeAny?          string
        |     |     +--rw excludeAny?          string
        |     |     +--rw hopLimit?            uint32
        |     |     +--ro lspId?               uint32
        |     |     +--ro lspState?            enumeration
        |     |     +--ro modifyLspId?         uint32
        |     +--rw resvStyle?                    enumeration
        |     +--rw tieBreaking?                  enumeration
        |     +--rw pathMetricType?               enumeration
        |     +--ro hotStandbySwitchReason?       enumeration
        |     +--ro adminStatus?                  enumeration
        |     +--ro operStatus?                   enumeration
        |     +--ro workingLspType?               enumeration
        |     +--ro workingLspId?                 uint32
        |     +--rw frrAttr
        |     |  +--rw frrEnable?          boolean
        |     |  +--rw bwProtEnable?       boolean
        |     |  +--rw frrBandwidth?       uint32
        |     |  +--rw frrSetupPriority?   uint32
        |     |  +--rw frrHoldPriority?    uint32
        |     +--rw bypassAttr
        |        +--rw bypassEnable?       boolean
        |        +--rw bypassProtectIFs
        |           +--rw bypassProtectIF* [bypassProtectIFName]
        |              +--rw bypassProtectIFName    ifName
```

o tunnelName: a tunnel name is unique among all tunnels established on a network node.

- o mplsTunnelIngressLSRId: ingress LSR ID of the tunnel.

- o mplsTunnelEgressLSRId: egress LSR ID of the tunnel.

- o mplsTunnelIndex: Session ID of a tunnel.

- o mplsTunnelBandwidth: tunnel bandwidth.

- o mplsTeTunnelSetupPriority: tunnel setup priority.

- o holdPriority: tunnel holding priority.

- o hotStandbyEnable: specifies hot standby capability for protecting TE tunnels.

- o hsbRevertiveMode: hot standby revertive mode. There are two revert modes which are revertive or non-revertive.

- o hotStandbyWtr: time of waiting recovering back to primary LSP. When hot-standby backup is in use, after primary LSP restores, the traffic will switch to primary LSP after waiting some time instead of switching to primary LSP immediately. This is to avoid frequent switching between primary LSP and backup LSP caused by network flapping.

- o ordinaryEnable: specifies tunnel ordinary backup protection capability. When it is enabled and the primary LSP fails, a backup LSP that meets certain limits will be set up. Then the traffic on the primary LSP will be switched to the backup LSP.

- o bestEffortEnable: specifies best-effort path protection of tunnels. When best-effort path is enabled for a TE tunnel, and both active and standby LSP fail, an LSP will be set up in the best effort method.

- o disableCspf: disable CSPF of a tunnel.

- o tunnelPath: information of tunnel paths belong to specified tunnel. There can be maximum four tunnel paths with different path type coexisting. Every tunnel path has independent path constraint and is associated with one set up LSP.

- o pathType: path type of a tunnel path. The available options are primary(used by primary LSP), hot-standby(used by hot-standby backup LSP), ordinary(used by ordinary backup LSP), and best-effort(used by best-effort LSP).

- o explicitPathName: name of an explicit path which is used for the tunnel path.

- o includeAll: Administrative group attribute of an LSP (IncludeAll).
- o includeAny: Administrative group attribute of an LSP (IncludeAny).
- o excludeAny: Administrative group attribute of an LSP (ExcludeAny).
- o hopLimit: number of limit of hops in a tunnel path.
- o lspId: LSP ID of a tunnel path.
- o lspState: the state of LSP.
- o modifyLspId: modified LSP ID of a tunnel path.
- o resvStyle: Tunnel reservation styles. SE style: shared explicit style; FF: fixed filter style.
- o tieBreaking: routing rules for a tunnel with multiple equal-cost routes which can be random, least fill and most fill. Random: Select a link randomly. Least fill: Select the link with smallest bandwidth usage. Most fill: Select the link with biggest bandwidth usage. By default, routing rules are inherited from the global MPLS TE routing rules. If multiple paths meet certain limits, a path will be selected based on the preceding rules.
- o pathMetricType: referenced metric type of one link for calculating path when creating TE tunnels. The available options are DEFAULT, IGP and TE.
- o hotStandbySwitchReason: the reason of hot-standby LSP switch.
- o adminStatus: administrative state of a tunnel which is UP or Down.
- o operStatus: operation status of a tunnel which is UP or Down.
- o workingLspType: type of working LSP which is primary, hot-standby, ordinary or best-effort.
- o workingLspId: LSP ID of working LSP.
- o frrEnable: specifies fast reroute capability.
- o bwProtEnable: The tunnel with fast reroute capability requests bandwidth protection.
- o frrBandwidth: FRR-protection bandwidth requested by an active tunnel.

o frrSetupPriority: Setup priority of FRR protection tunnels.

o frrHoldPriority: holding priority of FRR protection tunnels.

o bypassEnable: bypass tunnel enabling or disabling.

o bypassProtectIFName: Specifies the name of an interface that can be protected by a tunnel enabled with the bypass function.

2.1.3. RSVP-TE LSP

By collecting the LSP information of RSVP-TE protocol from the I2RS agent I2RS client can monitor the RSVP-TE LSP.

This section describes the information model related to RSVP-TE LSP which is shown in the Yang high-level description and interprets the meaning of each element.

```
   +--ro rsvpTeLsps
   |  +--ro rsvpTeLsp* [mplsTunnelIngressLSRId mplsTunnelEgressLSRId mpls
SessionID mplsLSPID]
   |     +--ro mplsTunnelIngressLSRId         inet:ipv4-address
   |     +--ro mplsTunnelEgressLSRId          inet:ipv4-address
   |     +--ro mplsSessionID                  uint16
   |     +--ro mplsLSPID                      uint16
   |     +--ro tunnelName?                    string
   |     +--ro mplsTunnelRole?                enumeration
   |     +--ro incomingIfName?                string
   |     +--ro outgoingIfName?                string
   |     +--ro mplsTunnelSetupPrio?           uint8
   |     +--ro mplsTunnelHoldingPrio?         uint8
   |     +--ro mplsTunnelBandwidth?           uint32
   |     +--ro mplsTunnelEHops
   |     |  +--ro mplsTunnelEHop* [mplsTunnelEHopIndex]
   |     |     +--rw mplsTunnelEHopIndex       uint32
   |     |     +--rw mplsTunnelEHopIpAddr      inet:ipv4-address
   |     |     +--rw mplsTunnelEHopType?       enumeration
   |     |     +--rw mplsTunnelEHopAddrType?   enumeration
   |     +--ro mplsTunnelARHops
   |     |  +--ro mplsTunnelARHop* [mplsTunnelARHopIndex]
   |     |     +--ro mplsTunnelARHopIndex              uint32
   |     |     +--ro incommingType?                    boolean
   |     |     +--ro mplsTunnelARHopIpAddr?            inet:ipv4-address
   |     |     +--ro mplsTunnelARHopLabel?             uint32
   |     |     +--ro mplsTunnelLocalProtectInUse?      boolean
   |     |     +--ro mplsTunnelARHopLocalProtectType?  enumeration
   |     |     +--ro mplsTunnelARHopBwProt?            boolean
   |     +--ro mplsTunnelHopTableName?                 string
   |     +--ro mplsCHops
```

```
      |     |  +--ro mplsCHop* [mplsTunnelCRPathIndex mplsTunnelCRHopIndex]
      |     |     +--ro mplsTunnelCRPathIndex             uint32
      |     |     +--ro mplsTunnelCRHopIndex              uint32
      |     |     +--ro mplsTunnelCRHopIsInclude?         boolean
      |     |     +--ro mplsTunnelCRHopType?              enumeration
      |     |     +--ro mplsTunnelLocalProtectInUse?      boolean
      |     |     +--ro mplsTunnelCRHopAddrType?          enumeration
      |     |     +--ro mplsTunnelCRHopIpAddr?            inet:ipv4-address
      |     +--ro mplsTunnelLocalProtectEnable?           boolean
      |     +--ro mplsTunnelLocalProtectInUse?            enumeration
      |     +--ro bypassTunnelName?                       string
      |     +--ro mplsTunnelMergingPermitted?             boolean
      |     +--ro mplsTunnelIncludeAllAffinity?           string
      |     +--ro mplsTunnelIncludeAnyAffinity?           string
      |     +--ro mplsTunnelExcludeAnyAffinity?           string
      |     +--ro lspMTU?                                 uint32
      |     +--ro mplsTunnelOperStatus?                   enumeration
      |     +--ro inLabel?                                uint32
      |     +--ro outLabel?                               uint32
      |     +--ro nextHop?                                inet:ipv4-address
      |     +--ro xcindex?                                uint32
```

o mplsTunnelIngressLSRId: ingress LSR ID of the tunnel.

o mplsTunnelEgressLSRId: egress LSR ID of the tunnel.

o mplsSessionID: session ID of a tunnel.

o mplsLSPID: LSP ID of a tunnel path.

o tunnelName: a tunnel name is unique among all tunnels established on a network node.

o mplsTunnelRole: the types of the LSP nodes, including ingress, transit, and egress nodes.

o incomingIfName: the name of incoming interface.

o outgoingIfName: the name of outgoing interface.

o mplsTunnelSetupPrio: setup priority of an LSP.

o mplsTunnelHoldingPrio: hold priority of an LSP.

o mplsTunnelBandwidth: bandwidth of a tunnel.

o mplsTunnelEHop: explicit path information included in Explicit Route Object.

- o mplsTunnelEHopIndex: hop index of an explicit path.

- o mplsTunnelEHopIpAddr: IP address of one hop.

- o mplsTunnelEHopType: an LSP route selection types based on the local hop which can be strict, loose and excluding. Strict type: Only an LSP route that includes the local hop can be selected. Loose type: An LSP route that includes the local node is selected preferentially. If the local hop does not meet path limits, it will be not included in the selected route. Excluding type: Only an LSP route that does not include the local hop can be selected.

- o mplsTunnelEHopAddrType: address type.

- o mplsTunnelARHop: actual path of an LSP.

- o mplsTunnelARHopIndex: hop index of actual path.

- o incommingType: specify whether the hop is an inbound interface.

- o mplsTunnelARHopIpAddr: IP address of the actual hop.

- o mplsTunnelARHopLabel: Label of the actual hop.

- o mplsTunnelLocalProtectInUse: FRR protection state.

- o mplsTunnelARHopLocalProtectType: FRR protection type.

- o mplsTunnelARHopBwProt: FRR bandwidth protection.

- o mplsTunnelHopTableName: explicit path name of an LSP.

- o mplsCHop: path calculated by CSPF according to LSP constraints.

- o mplsTunnelCRPathIndex: index of path calculated by CSPF.

- o mplsTunnelCRHopIndex: index of hop of path calculated by CSPF.

- o mplsTunnelCRHopIsInclude: specify if it is this hop a include hop.

- o mplsTunnelCRHopType: hop type calculated by CSPF. The available options are strict and loose.

- o mplsTunnelLocalProtectInUse: FRR protection state.

- o mplsTunnelCRHopAddrType: address type of hop of path calculated by CSPF. The available options are IPv4 and IPv6.

o mplsTunnelCRHopIpAddr: IP address of hop of path calculated by CSPF.

o mplsTunnelLocalProtectEnable: specifies the enabling or disabling state of FRR for an LSP.

o mplsTunnelLocalProtectInUse: specifies the FRR protection state of this LSP.

o bypassTunnelName: name of the bypass tunnel that protects the LSP.

o mplsTunnelMergingPermitted: specify whether the LSP permits bandwidth sharing.

o mplsTunnelIncludeAllAffinity: specifies the Include-all (administrative group attribute) of an LSP.

o mplsTunnelIncludeAnyAffinity: specifies the Include-any (administrative group attribute) of an LSP.

o mplsTunnelExcludeAnyAffinity: specifies the Exclude-any (administrative group attribute) of an LSP.

o lspMTU: specifies an LSP MTU.

o mplsTunnelOperStatus: operation status of an LSP.

o inLabel: incoming label of the transit or egress LSP.

o outLabel: out label of the ingress or transit LSP.

o nextHop: next hop address of the ingress or transit LSP.

o xcindex: LSP index of the LSP.

2.1.4. Static Tunnel

Network programming software managing the static CR-LSP devices may incorporate an I2RS Client along with a path calculation entity, a label management entity, and a bandwidth management entity. The I2RS Client should be able to communicate the static configuration to the network nodes, and monitor the status of the CR-LSPs. The static tunnel supports primary LSP and standby LSP.

This section describes the information model related to information of static tunnel which is shown in the Yang high-level description and interprets the meaning of each element.

```
        +--rw staticTunnels
        |  +--rw staticTunnel* [tunnelName]
        |     +--rw tunnelName                    string
        |     +--rw tunnelIngressLSRId?           inet:ipv4-address
        |     +--rw tunnelEgressLSRId?            inet:ipv4-address
        |     +--rw tunnelIndex?                  uint16
        |     +--rw tunnelBandwidth?              uint32
        |     +--rw staticCRLSPs
        |     |  +--rw staticCRLSP* [lspName]
        |     |     +--rw lspType                 enumeration
        |     |     +--rw lspName                 string
        |     |     +--ro lspState                enumeration
        |     +--rw revertMode?                   uint32
        |     +--rw wtrValue?                     uint32
        |     +--rw holdoffValue?                 uint32
```

o tunnelName: a static tunnel name is unique among all tunnels established on a network node.

o tunnelIngressLSRId: ingress LSR ID of the static tunnel.

o tunnelEgressLSRId: egress LSR ID of the static tunnel.

o tunnelIndex: session ID of the static tunnel.

o tunnelBandwidth: bandwidth of the static tunnel.

o staticCRLSP: static CR-LSP belonging to the same static tunnel

o lspType: type of static CR-LSP, working LSP or protection LSP.

o lspName: name of static CR-LSP.

o lspState: state of static CR-LSP.

o revertMode: two modes, one is non-revertive that traffic do not switch to primary LSP when primary LSP is up and the other is revertive that traffic switch to primary LSP when primary LSP is up.

o wtrValue: wait-to-restore time of traffic switch to primary LSP when primary LSP is up.

o holdoffValue: hold off time of traffic switch to standby LSP when the primary LSP is down.

2.1.5. Static CR-LSP

The static CR-LSP includes the information configured statically of the label, bandwidth, out interface etc. The LSP information downloaded to i2rs agent is different according to different role.

This section describes the information model related to static CR-LSP which is shown in the Yang high-level description and interprets the meaning of each element.

```
      +--rw staticCRLsps
      |  +--rw staticCRLsp* [lspName]
      |     +--rw lspName                    string
      |     +--rw lsrRole                    enumeration
      |     +--ro destinationAddress?        inet:ipv4-address
      |     +--rw bandwitdh                  uint32
      |     +--ro incomingIfName?            ifName
      |     +--rw inLabel?                   uint32
      |     +--rw outgoingIfName?            ifName
      |     +--rw nextHop?                   inet:ipv4-address
      |     +--rw outLabel?                  uint32
      |     +--rw mtu?                       uint32
```

o lspName: name of static CR-LSP.

o lsrRole: role of LSP which is ingress, transit and egress.

o destinationAddress: address of destination tunnel.

o bandwitdh: bandwidth of the static CR-LSP.

o incomingIfName: name of incoming interface for transit or egress LSP.

o inLabel: incoming label of transit or egress LSP.

o outgoingIfName: name of outgoing interface for ingress or transit LSP.

o nextHop: next hop of ingress or transit LSP.

o outLabel: out label of ingress or transit LSP.

o mtu: mtu of static LSP.

3. I2RS YANG model of MPLS TE

```
module: i2rs-mplste
   +--rw mplsTe
      +--rw explicitPaths
      |  +--rw explicitPath* [explicitPathName]
      |     +--rw explicitPathName    string
      |     +--rw explicitPathHops
      |        +--rw explicitPathHop* [mplsTunnelHopIndex]
      |           +--rw mplsTunnelHopIndex       uint32
      |           +--rw mplsTunnelHopIpAddr      inet:ipv4-address
      |           +--rw mplsTunnelHopType?       enumeration
      |           +--rw mplsTunnelHopAddrType?   enumeration
      +--rw rsvpTeTunnels
      |  +--rw rsvpTeTunnel* [tunnelName]
      |     +--rw tunnelName                    string
      |     +--ro mplsTunnelIngressLSRId?       inet:ipv4-address
      |     +--rw mplsTunnelEgressLSRId?        inet:ipv4-address
      |     +--rw mplsTunnelIndex?              uint16
      |     +--rw mplsTunnelBandwidth?          uint32
      |     +--rw mplsTeTunnelSetupPriority?    uint8
      |     +--rw holdPriority?                 uint8
      |     +--rw hotStandbyEnable?             boolean
      |     +--rw hsbRevertiveMode?             enumeration
      |     +--rw hotStandbyWtr?                uint32
      |     +--rw ordinaryEnable?               boolean
      |     +--rw bestEffortEnable?             boolean
      |     +--rw disableCspf?                  boolean
      |     +--rw tunnelPaths
      |     |  +--rw tunnelPath* [pathType]
      |     |     +--rw pathType             enumeration
      |     |     +--rw explicitPathName?    string
      |     |     +--ro includeAll?          string
      |     |     +--rw includeAny?          string
      |     |     +--rw excludeAny?          string
      |     |     +--rw hopLimit?            uint32
      |     |     +--ro lspId?               uint32
      |     |     +--ro lspState?            enumeration
      |     |     +--ro modifyLspId?         uint32
      |     +--rw resvStyle?                    enumeration
      |     +--rw tieBreaking?                  enumeration
      |     +--rw pathMetricType?               enumeration
      |     +--ro hotStandbySwitchReason?       enumeration
      |     +--ro adminStatus?                  enumeration
      |     +--ro operStatus?                   enumeration
      |     +--ro workingLspType?               enumeration
      |     +--ro workingLspId?                 uint32
      |     +--rw frrAttr
```

```
      |     |  +--rw frrEnable?          boolean
      |     |  +--rw bwProtEnable?       boolean
      |     |  +--rw frrBandwidth?       uint32
      |     |  +--rw frrSetupPriority?   uint32
      |     |  +--rw frrHoldPriority?    uint32
      |     +--rw bypassAttr
      |        +--rw bypassEnable?       boolean
      |        +--rw bypassProtectIFs
      |           +--rw bypassProtectIF* [bypassProtectIFName]
      |              +--rw bypassProtectIFName    ifName
      +--ro rsvpTeLsps
      |  +--ro rsvpTeLsp* [mplsTunnelIngressLSRId mplsTunnelEgressLSRId mpls
SessionID mplsLSPID]
      |     +--ro mplsTunnelIngressLSRId         inet:ipv4-address
      |     +--ro mplsTunnelEgressLSRId          inet:ipv4-address
      |     +--ro mplsSessionID                  uint16
      |     +--ro mplsLSPID                      uint16
      |     +--ro tunnelName?                    string
      |     +--ro mplsTunnelRole?                enumeration
      |     +--ro incomingIfName?                string
      |     +--ro outgoingIfName?                string
      |     +--ro mplsTunnelSetupPrio?           uint8
      |     +--ro mplsTunnelHoldingPrio?         uint8
      |     +--ro mplsTunnelBandwidth?           uint32
      |     +--ro mplsTunnelEHops
      |     |  +--ro mplsTunnelEHop* [mplsTunnelEHopIndex]
      |     |     +--rw mplsTunnelEHopIndex       uint32
      |     |     +--rw mplsTunnelEHopIpAddr      inet:ipv4-address
      |     |     +--rw mplsTunnelEHopType?       enumeration
      |     |     +--rw mplsTunnelEHopAddrType?   enumeration
      |     +--ro mplsTunnelARHops
      |     |  +--ro mplsTunnelARHop* [mplsTunnelARHopIndex]
      |     |     +--ro mplsTunnelARHopIndex              uint32
      |     |     +--ro incommingType?                    boolean
      |     |     +--ro mplsTunnelARHopIpAddr?            inet:ipv4-address
      |     |     +--ro mplsTunnelARHopLabel?             uint32
      |     |     +--ro mplsTunnelLocalProtectInUse?      boolean
      |     |     +--ro mplsTunnelARHopLocalProtectType?  enumeration
      |     |     +--ro mplsTunnelARHopBwProt?            boolean
      |     +--ro mplsTunnelHopTableName?                 string
      |     +--ro mplsCHops
      |     |  +--ro mplsCHop* [mplsTunnelCRPathIndex mplsTunnelCRHopIndex]
      |     |     +--ro mplsTunnelCRPathIndex             uint32
      |     |     +--ro mplsTunnelCRHopIndex              uint32
      |     |     +--ro mplsTunnelCRHopIsInclude?         boolean
      |     |     +--ro mplsTunnelCRHopType?              enumeration
      |     |     +--ro mplsTunnelLocalProtectInUse?      boolean
      |     |     +--ro mplsTunnelCRHopAddrType?          enumeration
      |     |     +--ro mplsTunnelCRHopIpAddr?            inet:ipv4-address
```

```
|     +--ro mplsTunnelLocalProtectEnable?          boolean
|     +--ro mplsTunnelLocalProtectInUse?           enumeration
|     +--ro bypassTunnelName?                      string
|     +--ro mplsTunnelMergingPermitted?            boolean
|     +--ro mplsTunnelIncludeAllAffinity?          string
|     +--ro mplsTunnelIncludeAnyAffinity?          string
|     +--ro mplsTunnelExcludeAnyAffinity?          string
|     +--ro lspMTU?                                uint32
|     +--ro mplsTunnelOperStatus?                  enumeration
|     +--ro inLabel?                               uint32
|     +--ro outLabel?                              uint32
|     +--ro nextHop?                               inet:ipv4-address
|     +--ro xcindex?                               uint32
+--rw staticTunnels
|  +--rw staticTunnel* [tunnelName]
|     +--rw tunnelName                  string
|     +--rw tunnelIngressLSRId?         inet:ipv4-address
|     +--rw tunnelEgressLSRId?          inet:ipv4-address
|     +--rw tunnelIndex?                uint16
|     +--rw tunnelBandwidth?            uint32
|     +--rw staticCRLSPs
|     |  +--rw staticCRLSP* [lspName]
|     |     +--rw lspType               enumeration
|     |     +--rw lspName               string
|     |     +--ro lspState              enumeration
|     +--rw revertMode?                 uint32
|     +--rw wtrValue?                   uint32
|     +--rw holdoffValue?               uint32
+--rw staticCRLsps
|  +--rw staticCRLsp* [lspName]
|     +--rw lspName                     string
|     +--rw lsrRole                     enumeration
|     +--ro destinationAddress?         inet:ipv4-address
|     +--rw bandwitdh                   uint32
|     +--ro incomingIfName?             ifName
|     +--rw inLabel?                    uint32
|     +--rw outgoingIfName?             ifName
|     +--rw nextHop?                    inet:ipv4-address
|     +--rw outLabel?                   uint32
|     +--rw mtu?                        uint32
```

Figure 2 The I2RS YANG model of MPLS TE

4. IANA Considerations

This draft includes no request to IANA.

5. Security Considerations

This document introduces no new security threat and SHOULD follow the security requirements as stated in [I-D.ietf-i2rs-architecture].

6. Normative References

[I-D.hares-i2rs-usecase-reqs-summary]
Hares, S., "Summary of I2RS Use Case Requirements", draft-hares-i2rs-usecase-reqs-summary-00 (work in progress), July 2014.

[I-D.huang-i2rs-mpls-te-usecases]
Huang, T., Li, Z., and S. Hares, "Use Cases for an Interface to MPLS TE", draft-huang-i2rs-mpls-te-usecases-02 (work in progress), July 2014.

[I-D.ietf-i2rs-architecture]
Atlas, A., Halpern, J., Hares, S., Ward, D., and T. Nadeau, "An Architecture for the Interface to the Routing System", draft-ietf-i2rs-architecture-05 (work in progress), July 2014.

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

Authors' Addresses

Xia Chen
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: jescia.chenxia@huawei.com

Zhenbin Li
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: lizhenbin@huawei.com

Susan Hares
Huawei Technologies
Saline, MI 48176
US

Email: shares@ndzh.com

Russ White
Ericsson
US

Email: russ.white@ericsson.com

Jeff Tantsura
Ericsson

Email: jeff.tantsura@ericsson.com