

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: February 16, 2015

X. Chen  
Z. Li  
X. Zeng  
Huawei Technologies  
August 15, 2014

Yang Model for MPLS Traffic Engineering(TE)  
draft-chen-mpls-te-yang-cfg-00

Abstract

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

Requirements Language

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

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on February 16, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. Terminology . . . . .	2
3. Design of Data Model . . . . .	3
3.1. Overview . . . . .	3
3.2. MPLS TE Global Configuration . . . . .	4
3.3. MPLS TE Link Configuration . . . . .	4
3.4. Explicit Path Configuration . . . . .	5
3.5. P2MP TE Leaf List Configuration . . . . .	5
3.6. RSVP-TE Tunnel Configuration . . . . .	5
3.7. RSVP-TE Global Configuration . . . . .	7
3.8. RSVP-TE Interface Configuration . . . . .	7
3.9. CSPF Configuration . . . . .	8
3.10. P2MP TE Tunnel Template Configuration . . . . .	8
4. MPLS TE Yang Module . . . . .	9
5. IANA Considerations . . . . .	37
6. Security Considerations . . . . .	37
7. Acknowledgements . . . . .	37
8. Normative References . . . . .	37
Authors' Addresses . . . . .	37

## 1. Introduction

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

This document defines a YANG data model that can be used to configure and manage MPLS TE. Both P2P TE and P2MP TE are supported.

## 2. Terminology

TE: Traffic Engineering

P2MP TE: Point-to-Multipoint Traffic Engineering

FRR: Fast Re-Route

SRLG: Shared Risk Link Group

CSPF: Constrained Shortest Path First

### 3. Design of Data Model

#### 3.1. Overview

The MPLS TE Yang module is divided in for main containers :

- o `mplsTeSite`: that contains global writable configuration objects for MPLS TE.

- o `teLinks`: that contains writable configuration objects for MPLS TE link.

- o `explicitPaths` : that contains writable configuration objects for explicit path used for MPLS TE tunnel.

- o `p2mpLeafLists` : that contains writable configuration objects for the list of leaf nodes of P2MP TE tunnel.

- o `rsvpTeTunnels` : that contains writable configuration objects for RSVP-TE tunnels.

- o `rsvpTESite` : that contains writable configuration objects for RSVP-TE.

- o `rsvpInterfaces` : that contains writable configuration objects for RSVP-TE.

- o `cspfCfg` : that contains writable configuration objects for CSPF.

- o `p2mpTeTemplates`: that contains writable configuration objects for P2MP TE tunnel template which is used for setting up P2MP TE tunnel triggered by multicast services.

When implement MPLS TE features, it is necessary to flood MPLS TE link information through IGP signaling including OSPF-TE and ISIS-TE. The Yang configuration of ISIS-TE and OSPF-TE is out of scope of this document.

The figure below describe the overall structure of the MPLS TE Yang module :

```

module: mplste
  +--rw mplste
    +--rw mplsteSite
      | ...
    +--rw teLinks
      | ...
    +--rw explicitPaths
      | ...
    +--rw p2mpLeafLists
      | ...
    +--rw rsvpTeTunnels
      | ...
    +--rw rsvpTESite
      | ...
    +--rw rsvpInterfaces
      | ...
    +--rw cspfCfg
      | ...
    +--rw p2mpTeTemplates
      | ...

```

### 3.2. MPLS TE Global Configuration

MPLS TE global configuration includes global parameters to control the flooding the TE link information, LSP switch auto FRR, auto bandwidth adjustment.

```

  +--rw mplsteSite
    | +--rw enablePeriFl?      boolean
    | +--rw floodingInterval? uint32
    | +--rw switchDelay?      uint32
    | +--rw autoFrrEnable?    boolean
    | +--rw autobwEnable?     boolean
    | +--rw autobwInterval?   uint32

```

### 3.3. MPLS TE Link Configuration

MPLS TE link configuration includes the parameters of the MPLS links such as bandwidth, administration group, SRLG, TE metric, auto FRR mode and the threshold value to control the flooding of MPLS TE link information.

```

+--rw teLinks
|   +--rw teLink* [interfaceName]
|       +--rw interfaceName          ifName
|       +--rw teIfMaxreservablebandwidth? uint32
|       +--rw teIfBc0bandwidth?      uint32
|       +--rw adminGroups?           string
|       +--rw srlgGroups
|       |   +--rw srlgGroup* [teIfSrlgValue]
|       |   |   +--rw teIfSrlgValue    uint32
|       +--rw bwChangeThresholdDown?  uint32
|       +--rw bwChangeThresholdUp?    uint32
|       +--rw teIfMetric?              uint32
|       +--rw autoFrrMode?             enumeration

```

### 3.4. Explicit Path Configuration

Explicit path configuration includes the list of IP addresses of strict or loose hops of the explicit path.

```

+--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

```

### 3.5. P2MP TE Leaf List Configuration

P2MP TE leaf list configurations includes the list of leaf nodes of the P2MP TE tunnel and the explicit paths used for these leaf nodes.

```

+--rw p2mpLeafLists
|   +--rw p2mpLeafList* [leafListName]
|       +--rw leafListName    string
|       +--rw leafs
|       |   +--rw leaf* [leafIpAddr]
|       |   |   +--rw leafIpAddr    inet:ipv4-address
|       |   |   +--rw explicitPathName? string

```

### 3.6. RSVP-TE Tunnel Configuration

RSVP-TE tunnel configuration includes the parameters for the RSVP-TE tunnel and the parameters for different LSPs (primary LSP, hot-standby LSP, ordinary backup LSP and best-effort LSP) of the RSVP-TE tunnels.

```

+--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
|   |   +--rw resvStyle?                   enumeration
|   |   +--rw mplsTunnelRecordRoute?        enumeration
|   |   +--rw reoptimization?               boolean
|   |   +--rw reoptiFrequency?              uint32
|   |   +--rw tieBreaking?                  enumeration
|   |   +--rw pathMetricType?               enumeration
|   |   +--rw AutoBandwidths
|   |   |   +--rw AutoBandwidth
|   |   |   |   +--rw AutoBwMode?           enumeration
|   |   |   |   +--rw thresholdPerc?        uint32
|   |   |   |   +--rw AutoBwFreq?           uint32
|   |   |   |   +--rw AutoBwMax?            uint32
|   |   |   |   +--rw AutoBwMin?            uint32
|   |   +--ro adminStatus?                  enumeration
|   |   +--ro operStatus?                    enumeration
|   |   +--rw tunnelInterface
|   |   |   +--rw statEnable?    boolean
|   |   |   +--rw igpAttr
|   |   |   |   +--rw advertiseEnable?        boolean
|   |   |   |   +--rw shortcutType?           enumeration
|   |   |   |   +--rw igpMetricType?          enumeration
|   |   |   |   +--rw relativeIgpMetricValue? int16
|   |   |   |   +--rw absoluteIgpMetricValue? uint16
|   |   |   |   +--rw advertiseHoldTime?      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

```

### 3.7. RSVP-TE Global Configuration

RSVP-TE global configuration includes the global parameters for the RSVP-TE signaling such as different timer parameters, hello capability, SRefresh capability, etc.

```

+--rw rsvpTESite
|   +--rw helloEnable?           boolean
|   +--rw maxHelloMissTimes?     uint32
|   +--rw helloInterval?        uint32
|   +--rw supportGREnable?      boolean
|   +--rw keepMultiplier?       uint32
|   +--rw refreshInterval?      uint32
|   +--rw srefreshEnable?       boolean
|   +--rw retransmissionInterval? uint32
|   +--rw retransmissionIncrementValue? uint32
|   +--rw challengeRetransmissionInterval? uint32
|   +--rw maxChallengeMissTimes? uint32

```

### 3.8. RSVP-TE Interface Configuration

RSVP-TE interface configuration includes the parameter of the RSVP-TE interface such as interface name, hello capability, auto FRR mode and authentication.

```

+--rw rsvpInterfaces
|   +--rw rsvpInterface* [interfaceName]
|       +--rw interfaceName    ifName
|       +--rw helloEnable?     boolean
|       +--rw autoFrrMode?     enumeration
|       +--rw authentication
|           +--rw authEnable?   boolean
|           +--rw authMD5Key?   string
|           +--rw authLifetime? uint32
|           +--rw authHandshake? string
|           +--rw authWindowSize? uint32

```

### 3.9. CSPF Configuration

CSPF configuration includes the IGP choice, tie-breaking policy, metric type, SRLG policy for the MPLS TE path calculation.

```

+--rw cspfCfg
|   +--rw enableCspf?          boolean
|   +--rw preferredIgp?        enumeration
|   +--rw preferredOspfProcessId? uint32
|   +--rw preferredOspfAreaId?  boolean
|   +--rw preferredIsisProcessId? uint32
|   +--rw preferredIsisLevel?   enumeration
|   +--rw tiebreaking?          enumeration
|   +--rw pathMetricType?       enumeration
|   +--rw srlgPathCalcMode?     enumeration

```

### 3.10. P2MP TE Tunnel Template Configuration

P2MP TE tunnel template will be used to set up P2MP TE tunnel triggered by multicast service such as BGP-base MVPN defined in [RFC6514].

```

+--rw p2mpTeTemplates
|   +--rw p2mpTeTemplate* [templateName]
|       +--rw templateName          string
|       +--rw recordRouteMode?       enumeration
|       +--rw resvStyle?              enumeration
|       +--rw setupPriority?           uint8
|       +--rw holdPriority?            uint8
|       +--rw bandwidth?              uint32
|       +--rw reoptimization?         boolean
|       +--rw reoptiFrequency?        uint32
|       +--rw pathMetricType?         enumeration
|       +--rw tieBreaking?            enumeration
|       +--rw hopLimit?               uint32
|       +--rw includeAllAffinity?     string
|       +--rw includeAnyAffinity?     string
|       +--rw excludeAnyAffinity?     string
|       +--rw leafListName            string
|       +--rw mplsTeP2mpTemplateFrr
|           +--rw frrEnable?          boolean
|           +--rw bwProtEnable?       boolean
|           +--rw frrBandwidth?       uint32
|           +--rw frrSetupPriority?    uint32
|           +--rw frrHoldPriority?     uint32

```



## 4. MPLS TE Yang Module

```
module mplste {
    namespace "urn:huawei:params:xml:ns:yang:mplste";
    // replace with IANA namespace when assigned - urn:ietf:params:xml:ns:yang:1
    prefix "mplste";
    import ietf-inet-types {
        prefix inet;
    }
    organization
        "Huawei Technologies Co., Ltd.";
    contact
        "jescia.chenxia@huawei.com
        lizhenbin@huawei.com
        zengxinzong@huawei.com";
    description
        "This YANG module defines the generic configuration
        data for MPLS TE, which is common across all of the vendor
        implementations of the protocol. It is intended that the module
        will be extended by vendors to define vendor-specific
        MPLS TE configuration parameters.";
    revision 2014-08-16 {
        description
            "Initial revision.";
    }

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

    container mplste {

        container mplsteSite {

            description "MPLS TE basic Configuration.";

            leaf enablePeriFl {
                description "Capability of flooding TE link bandwidth periodical
                ly. When TE LSP reserves or releases bandwidth with the changing rate not reachi
                ng the flooding threshold of TE link, it will not flood bandwidth to network. In
                order to flood bandwidth to network as soon as possible, and to avoid waste of
                network resource caused by frequent flooding, it will flood changed bandwidth to
                network periodically.";
                config "true";
                default "false";
                type boolean;
            }
            leaf floodingInterval {
                description "Interval at which a TE interface floods bandwidth o
                ver the network. The interval ranges from 10s to 43200s, and the default value i
                s 30s. The bandwidth that a TE LSP reserves or releases on an interface may be o
                nly a small part compared with the reservable bandwidth of the interface, which
                does not reach the flooding threshold of the interface. To flood the interface b
                andwidth over the network and avoid frequent flooding, the TE NE is configured t
                o flood bandwidth over the network periodically.";
                config "true";
            }
        }
    }
}
```



```

        default "30";
        type uint32 {
            range "10..43200";
        }
    }
    leaf switchDelay {
        description "Delay time for switching the TE traffic from a Primary CR-LSP to a Modified CR-LSP. The delay time ranges from 0s to 120000s, and the default delay value is 5000s.";
        config "true";
        default "5000";
        type uint32 {
            range "0..120000";
        }
    }
    leaf autoFrrEnable {
        description "Specifies the enabling state of auto FRR.";
        config "true";
        default "false";
        type boolean;
    }
    leaf autobwEnable {
        description "Specifies the enabling state of auto bandwidth.";
        config "true";
        default "false";
        type boolean;
    }
    leaf autobwInterval {
        description "Set interval of Auto bandwidth sampling.";
        config "true";
        default "300";
        type uint32 {
            range "1..604800";
        }
    }
}

container teLinks {

    list teLink {

        key "interfaceName";
        max-elements "unbounded";
        min-elements "0";
        description "TELINK";

        leaf interfaceName {
            description "Specifies the name of a physical interface where TE is enabled.";
            config "true";
            type ifName;
        }
    }
}

```

```

    }
    leaf teIfMaxreservablebandwidth {
        description "Specifies the maximum reserved bandwidth (kbit/
s) of TE interfaces. The value ranges from 0 kbit/s to 4294967295 kbit/s. By def
ault, the value is 0 Kbit/s. The limit of interface bandwidth is as follows: The
configured bandwidth on a TE interface cannot exceed the physical bandwidth of
the interface. Otherwise, the NE prompts an error.";
        config "true";
        default "0";
        type uint32 {
            range "0..4294967295";
        }
    }
    leaf teIfBc0bandwidth {
        description "Bc0 bandwidth value in kbps";
        config "true";
        default "0";
        type uint32 {
            range "0..4294967295";
        }
    }
    leaf adminGroups {
        description "Specifies interface management attributes. They
are total 32 bits. Each bit is one attribute that is established on a TE LSP. S
tandby LSPs are selected based on the limit of the LSP affinity attribute.
It's hexadecimal string.";
        config "true";
        default "00000000";
        type string {
            length "8";
        }
    }
    container srlgGroups {
        list srlgGroup {
            key "teIfSrlgValue";
            max-elements "unbounded";
            min-elements "0";
            description "Specifies a risk-shared link group.";

            leaf teIfSrlgValue {
                description "Specifies a risk-shared link group numb
er.";
                config "true";
                type uint32 {
                    range "0..4294967295";
                }
            }
        }
    }

    leaf bwChangeThresholdDown {
        description "Specifies the flooding threshold of bandwidth c
onsumption percentage. The value ranges from 0?? to 100??. By default, the value
is 10??. Establishing TE LSPs consumes the bandwidth of the TE interface, and t
he available bandwidth of the interface is therefore reduced. If the percentage
of the reduced bandwidth is larger than the configured flooding threshold, the b
andwidth of the TE interface will be flooded to the network.";

```



```

        config "true";
        default "10";
        type uint32 {
            range "0..100";
        }
    }
    leaf bwChangeThresholdUp {
        description "Specifies the flooding threshold of the percent
age of released bandwidth compared with the available interface bandwidth. The v
alue ranges from 0?? to 100??. By default, the value is 10??. When a TE LSP of a
n interface is removed, the bandwidth of the TE LSP is released. The available b
andwidth of the interface is therefore increased. If the percentage of the incre
ased bandwidth is larger than the configured flooding threshold, the bandwidth o
f the TE interface will be flooded over the network.";
        config "true";
        default "10";
        type uint32 {
            range "0..100";
        }
    }
    leaf teIfMetric {
        description "Specifies the Value of TE interface metric. The
value ranges from 1 to 16777215. By default, the value is 1. The link with smal
lest metric value is used preferentially as a TE LSP.";
        config "true";
        type uint32 {
            range "1..16777215";
        }
    }
    leaf autoFrrMode {
        description "Auto FRR protection modes: global inheritance,
link protection, node protection and self-adapting. By default, global AutoFRR i
s global inheritance. Link protection: A bypass tunnel can be used to protect on
ly the links where the outbound interface of the primary LSP resides. Node prote
ction: A bypass tunnel can be used to protect the downstream node of the links w
here the outbound interface of the primary LSP resides. Self-adapting??A bypass t
unnel can be dynamically selected node protection or link protection according t
o the network conditions.";
        config "true";
        default "DEFAULT";
        type enumeration {
            enum DEFAULT {
                value "0";
                description "DEFAULT:";
            }
            enum LINK {
                value "1";
                description "LINK:";
            }
            enum NODE {
                value "2";
                description "NODE:";
            }
            enum DISABLE {
                value "3";
                description "DISABLE:";
            }
        }
    }
}

container explicitPaths {

```



```

list explicitPath {

    key "explicitPathName";
    max-elements "unbounded";
    min-elements "0";
    description "Explicit path.";

    leaf explicitPathName {
        description "Name of an explicit path.";
        config "true";
        type string {
            length "1..31";
            pattern "^[^ \?]*$";
        }
    }
}
container explicitPathHops {

    list explicitPathHop {

        key "mplsTunnelHopIndex";
        max-elements "unbounded";
        min-elements "0";

        leaf mplsTunnelHopIndex {
            description "Hop index of an explicit path.";
            config "true";
            type uint32 {
                range "1..65535";
            }
        }
        leaf mplsTunnelHopIpAddress {
            description "IP address of hop.";
            config "true";
            mandatory "true";
            type inet:ipv4-address;
        }
        leaf mplsTunnelHopType {
            description "Specifies an LSP route selection types
based on the local hop. Strict type: Only an LSP route that includes the local h
op can be selected. Loose type: An LSP route that includes the local node is sel
ected 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 no
t include the local hop can be selected.";
            config "true";
            default "includeStrict";
            type enumeration {
                enum includeStrict {
                    value "0";
                    description "Strictly included.";
                }
                enum includeLoose {
                    value "1";
                    description "Loosely included.";
                }
            }
        }
    }
}

```



```

        }
        enum exclude {
            value "2";
            description "Excluded.";
        }
    }
}
leaf mplsTunnelHopAddrType {
    description "Address type.";
    config "true";
    default "IPv4";
    type enumeration {
        enum IPV4 {
            value "0";
            description "IPv4:";
        }
    }
}
}
}
}
}
}
}
}
}
}
container p2mpLeafLists {
    list p2mpLeafList {
        key "leafListName";
        max-elements "unbounded";
        min-elements "0";
        description "Leaf list.";
        leaf leafListName {
            description "Name of leaf list.";
            config "true";
            type string {
                length "1..31";
                pattern "^[^ \\?]*$";
            }
        }
    }
    container leafs {
        list leaf {
            key "leafIpAddr";

```

```

        max-elements "unbounded";
        min-elements "0";
    description "Leaf.";

    leaf leafIpAddress {
        description "Leaf Address.";
        config "true";
        type inet:ipv4-address;
    }
    leaf explicitPathName {
        description "Explicit path name used for a leaf.";
        config "true";
        type string {
            length "0..31";
        }
    }
}

}

}

}

container rsvpTeTunnels {

    list rsvpTeTunnel {

        key "tunnelName";
        max-elements "unbounded";
        min-elements "0";
        description "TE E2E Dynamic unicast tunnel.";

        leaf tunnelName {
            description "Tunnel name. A tunnel name is unique among all
tunnels established on a node. Specifies a tunnel name whose prefix is Tunnel. T
unnel interfaces can be named in the following formats: 1. Tunnel x/y/z ('x' ran
ges from 0 to 16, 'y' ranges from 0 to 15, and 'z' ranges from 0 to 65535); 2.Tu
nnelN ('N' ranges from 0 to 4294967295).Under the environment of clusters, Tunne
l interfaces can be named in the following formats: 1. Tunnelx/y/z/k ('x' ranges
from 0 to 16, 'y' ranges from 0 to 16,'z' ranges from 0 to 15, and 'k' ranges f
rom 0 to 65535); 2.TunnelN ('N' ranges from 0 to 4294967295).";
            config "true";
            type string {
                length "0..63";
                pattern "^[^ ]*$";
            }
        }
        leaf mplsTunnelIngressLSRId {
            description "Specifies ingress LSR ID of the tunnel.";
            config "false";
            type inet:ipv4-address;
        }
        leaf mplsTunnelEgressLSRId {
            description "Specifies egress LSR ID of the tunnel.";

```

```

        config "true";
        type inet:ipv4-address;
    }
    leaf mplsTunnelIndex {
        description "Session ID of a tunnel.";
        config "true";
        type uint16;
    }
    leaf mplsTunnelBandwidth {
        description "Specifies a tunnel bandwidth. The value ranges
from 0 kbit/s to 4000000000 kbit/s. By default, the value is 0 kbit/s.";
        config "true";
        default "0";
        type uint32 {
            range "0..4000000000";
        }
    }
    leaf mplsTeTunnelSetupPriority {
        description "Specifies a tunnel setup priority. The value ra
nges from 0 to 7. By default, the value is 7. The smaller the value, the higher
the setup priority. 0 is the highest priority. Limit: the setup priority of a tu
nnel must be equal to or smaller than its holding priority.";
        config "true";
        default "7";
        type uint8 {
            range "0..7";
        }
    }
    leaf holdPriority {
        description "Specifies a tunnel holding priority. The value
ranges from 0 to 7. By default, the value is 7. The smaller the value, the highe
r the holding priority. 0 is the highest priority. Limit: The holding priority o
f a tunnel must be equal to or larger than its setup priority.";
        config "true";
        default "7";
        type uint8 {
            range "0..7";
        }
    }
    leaf hotStandbyEnable {
        description "Enabling of hot standby for protecting TE tunne
ls. When an active LSP is set up successfully, a standby LSP that meets certain
limits will be set up to protect the active LSP. When the active LSP fails, the
traffic on the active LSP will be switched to the standby LSP.";
        config "true";
        default "false";
        type boolean;
    }
    leaf hsbRevertiveMode {
        description "hot-standby revertive??There are two revert mod
es,e.g. revertive and non-revertive.The default mode is revertive.";
        config "true";
        default "revertive";
        type enumeration {
            enum revertive {
                value "0";
                description "revertive";
            }
            enum nonRevertive {

```



```

        value "1";
        description "non-evertive";
    }
}
}
leaf hotStandbyWtr {
    description "Time of waiting recovering back to primary LSP.
Its range is 0~2592000, default is 10. 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.";
    config "true";
    default "10";
    type uint32 {
        range "0..2592000";
    }
}
leaf ordinaryEnable {
    description "Specifies a tunnel ordinary backup protection c
apability. When it is enabled, and the primary LSP fails, a backup LSP that meet
s certain limits will be set up. Then, the traffic on the primary LSP will be sw
itched to the backup LSP. ";
    config "true";
    default "false";
    type boolean;
}
leaf bestEffortEnable {
    description "Best-effort path protection of tunnels. When be
st-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.";
    config "true";
    default "false";
    type boolean;
}
leaf disableCspf {
    description "disable Cspf of a tunnel";
    config "true";
    default "false";
    type boolean;
}
container tunnelPaths {
    list tunnelPath {
        key "pathType";
        max-elements "unbounded";
        min-elements "0";
        description "Path configuration of a tunnel.";

        leaf pathType {
            description "Path role of a tunnel. The available op
tions are primary(used by primary LSP), hot-standby(used by hot-standby backup L
SP), ordinary(used by ordinary backup LSP), and best-effort(used by best-effort
LSP).";
            config "true";
            type enumeration {
                enum primary {
                    value "0";
                    description "Primary path.";
                }
            }
        }
    }
}

```



```

        enum hot_standby {
            value "1";
            description "Hot standby path.";
        }
        enum ordinary {
            value "2";
            description "Ordinary backup path.";
        }
        enum best_effort {
            value "3";
            description "Best-effort path.";
        }
    }
}
leaf explicitPathName {
    description "Name of an explicit path.";
    config "true";
    type string {
        length "0..31";
    }
}
leaf includeAll {
    description "Administrative group attribute of an LS
P (includeAll).It's hexadecimal string.";
    config "false";
    default "00000000";
    type string {
        length "8";
    }
}
leaf includeAny {
    description "Administrative group attribute of an LS
P (includeAny).It's hexadecimal string.";
    config "true";
    default "00000000";
    type string {
        length "8";
    }
}
leaf excludeAny {
    description "Tunnel path management attribute: Exclu
de-any. The value of this attribute ranges from 0x0 to 0xFFFFFFFF. By default, t
he value is 0x0. The management attribute is a 32-bit vector. If the management
attribute of a link contains any bit in the Exclude-any attribute field of an LS
P, the link cannot be a candidate LSP link.It's hexadecimal string.";
    config "true";
    default "00000000";
    type string {
        length "8";
    }
}
leaf hopLimit {
    description "Number limit of hops in a tunnel path.
The value ranges from 1 to 32. By default, the value is 32.";
    config "true";

```

```

        default "32";
        type uint32 {
            range "1..32";
        }
    }
}

leaf resvStyle {
    description "Tunnel reservation styles. SE style: shared explicit style; FF: fixed filter style. The default tunnel reservation style is SE.";
    config "true";
    default "SE";
    type enumeration {
        enum SE {
            value "0";
            description "Shared Explicit Style:";
        }
        enum FF {
            value "1";
            description "Fixed Filter Style:";
        }
    }
}

leaf mplsTunnelRecordRoute {
    description "Route record mode of a tunnel. No-route record mode, route recording mode, and route and label record mode.";
    config "true";
    default "DISABLE";
    type enumeration {
        enum DISABLE {
            value "0";
            description "DISABLE:";
        }
        enum RECORD_ROUTE_ONLY {
            value "1";
            description "The LSP records routes only.";
        }
        enum RECORD_LABEL {
            value "2";
            description "The LSP records both routes and labels.";
        }
    }
}

leaf reoptimization {
    description "Auto reoptimization enabling state of TE Tunnel";
    config "true";
    default "false";
    type boolean;
}

```



```

    }
    leaf reoptiFrequency {
      description "Frequency of auto reoptimization, its range is
60-604800.";
      config "true";
      default "3600";
      type uint32 {
        range "60..604800";
      }
    }
    leaf tieBreaking {
      description "Routing rules for a tunnel with multiple equal-
cost routes. Random: Select a link randomly. least fill: Select the link with sm
allest 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 prec
eding rules.";
      config "true";
      default "DEFAULT";
      type enumeration {
        enum LEASTFILL {
          value "0";
          description "LEASTFILL:The link with the smallest ba
ndwidth occupation ratio is selected in the case of equal conditions.";
        }
        enum MOSTFILL {
          value "1";
          description "MOSTFILL:The link with the largest band
width occupation ratio is selected in the case of equal conditions.";
        }
        enum RANDOM {
          value "2";
          description "RANDOM:Links are selected randomly.";
        }
        enum DEFAULT {
          value "3";
          description "Inherit from global configuration.";
        }
      }
    }
    leaf pathMetricType {
      description "Referenced metric type of one link for calculat
ing path when creating TE tunnels. The available options are DEFAULT, IGP and TE
, default is inheriting from global configuration.";
      config "true";
      default "NONE";
      type enumeration {
        enum NONE {
          value "0";
          description "Inherit from global configuration.";
        }
        enum IGP {
          value "1";
          description "IGP";
        }
        enum TE {
          value "2";
          description "TE";
        }
      }
    }

```

```

    }
  }
}
container AutoBandwidths {
  container AutoBandwidth {
    description "Auto bandwidth.";

    leaf AutoBwMode {
      description "Auto bandwidth mode.";
      config "true";
      default "DISABLE";
      type enumeration {
        enum ADJUSTMENT {
          value "0";
          description "Adjustment mode??a lsp will be
created??if all of the conditions are satisfied.";
        }
        enum COLLECTBW {
          value "1";
          description "Collect mode??only collect band
width in this mode.";
        }
        enum DISABLE {
          value "2";
          description "Disable capability of Auto band
width.";
        }
      }
    }
  }
  leaf thresholdPerc {
    description "Set the threshold of the adjustment ban
dwidth.";

    config "true";
    default "0";
    type uint32 {
      range "0..100";
    }
  }
  leaf AutoBwFreq {
    description "Set Frequency of Auto bandwidth.";
    config "true";
    default "86400";
    type uint32 {
      range "300..604800";
    }
  }
  leaf AutoBwMax {
    description "Max Bandwidth.";
    config "true";
    default "4000000000";
  }
}

```

```
        type uint32 {
            range "0..4000000000";
        }
    }
    leaf AutoBwMin {
        description "Min Bandwidth.";
        config "true";
        default "0";
        type uint32 {
            range "0..4000000000";
        }
    }
}

leaf adminStatus {
    description "Administrative state of a tunnel--(UP??Down)";
    config "false";
    default "up";
    type enumeration {
        enum down {
            value "0";
            description "down:";
        }
        enum up {
            value "1";
            description "up:";
        }
    }
}

leaf operStatus {
    description "Operation status of a tunnel--(UP??Down)";
    config "false";
    type enumeration {
        enum down {
            value "0";
            description "down:";
        }
        enum up {
            value "1";
            description "up:";
        }
    }
}

container tunnelInterface {

    description "TE tunnel interface.";
```

```

    leaf statEnable {
        description "Traffic statistic enabling state.";
        config "true";
        type boolean;
    }
    container igpAttr {

        description "IGP attribute of tunnel interface.";

        leaf advertiseEnable {
            description "Tunnel interface forwarding adjacency b
ased on tunnel interfaces. An MPLE TE tunnel can be advertised as a virtual link
over an IGP network. The virtual link can participate in route calculation.";
            config "true";
            default "false";
            type boolean;
        }
        leaf shortcutType {
            description "Tunnel interface shortcut types. Disabl
ed: This function is not enabled, and virtual TE tunnel interfaces do not partic
ipate in route calculation. OSPF type: Virtual TE tunnel interfaces participate
in OSPF route calculation. ISIS type: Virtual TE tunnel interfaces participates
in ISIS route calculation. Limit: IGP Shortcut and forwarding adjacency cannot f
unction together.";
            config "true";
            default "disable";
            type enumeration {
                enum disable {
                    value "0";
                    description "disable:";
                }
                enum ospf {
                    value "1";
                    description "ospf:";
                }
                enum isis {
                    value "2";
                    description "isis:";
                }
                enum both {
                    value "3";
                    description "both:";
                }
            }
        }
        leaf igpMetricType {
            description "IGP metric types of tunnel interfaces.
Relative metric type and absolute metric type.";
            config "true";
            default "relative";
            type enumeration {
                enum absolute {
                    value "0";
                    description "absolute:";
                }
                enum relative {

```

```

        value "1";
        description "relative:";
    }
}
}
leaf relativeIgpMetricValue {
    description "IGP relative metric value of a tunnel i
nterface.";
    config "true";
    default "0";
    type int16 {
        range "-10..10";
    }
}
leaf absoluteIgpMetricValue {
    description "IGP absolute metric value of a tunnel i
nterface.";
    config "true";
    default "1";
    type uint16 {
        range "1..65535";
    }
}
leaf advertiseHoldTime {
    description "When Tunnel get down, the Time of Notif
ying tunnel as virtual link to other routers.";
    config "true";
    default "0";
    type uint32 {
        range "0..4294967295";
    }
}
}
}
container frrAttr {
    description "Fast reroute attribute.";
    leaf frrEnable {
        description "Request of fast reroute capability.";
        config "true";
        default "false";
        type boolean;
    }
    leaf bwProtEnable {
        description "The tunnel with fast reroute capability req
uests bandwidth protection.";
        config "true";
        default "false";
        type boolean;
    }
}

```

```

    }
    leaf frrBandwidth {
        description "FRR-protection bandwidth (kbits/s) request
ed by an active tunnel. The value ranges from 0 kbit/s to 4000000000 kbit/s. by
default, the value is 0 Kbit/s. The value cannot exceed the bandwidth of the act
ive tunnel.";
        config "true";
        default "0";
        type uint32 {
            range "0..4000000000";
        }
    }
    leaf frrSetupPriority {
        description "Setup priority of FRR-protection tunnels. T
he value ranges from 0 to 7. By default, the value is 7. The smaller the value,
the higher the setup priority. 0 is the highest priority. Limit: The protection
tunnel setup priority cannot exceed the setup priority of the active tunnel.";
        config "true";
        default "7";
        type uint32 {
            range "0..7";
        }
    }
    leaf frrHoldPriority {
        description "Holding priority of FRR protection tunnels.
The value ranges from 0 to 7. The smaller the value, the higher the priority. T
he value 0 is the highest priority. Limit: The protection tunnel holding priorit
y cannot exceed the active tunnel holding priority.";
        config "true";
        default "7";
        type uint32 {
            range "0..7";
        }
    }
}

container bypassAttr {

    description "Bypass tunnel attribute.";

    leaf bypassEnable {
        description "Bypass tunnel enabling or disabling. A bypa
ss tunnel can be enabled to protect a tunnel that requests FRR protection. Limit
: A bypass tunnel cannot request FRR protection.";
        config "true";
        default "false";
        type boolean;
    }
    container bypassProtectIFs {

        list bypassProtectIF {

            key "bypassProtectIFName";
            max-elements "unbounded";
            min-elements "0";
            description "Specifies a list of interfaces that can
be protected by a bypass tunnel.";

            leaf bypassProtectIFName {
                description "Specifies the name of an interface
that can be protected by a tunnel enabled with the bypass function.";

```



```

        config "true";
        type ifName;
    }
}

}

}

}

}

container rsvpTESite {

    description "RSVP configuration and status.";

    leaf helloEnable {
        description "Hello enabling state.";
        config "true";
        default "false";
        type boolean;
    }
    leaf maxHelloMissTimes {
        description "Maximum number of Hello packet loss times. The value ranges from 3 to 10, and the default value is 3. When the number of Hello packet refreshing (from the NE) times exceeds the configured maximum number, and the NE does not receive a response from its peer end, the NE considers that the session with the peer end is lost.";
        config "true";
        default "3";
        type uint32 {
            range "3..10";
        }
    }
    leaf helloInterval {
        description "Interval for refreshing Hello packets. It is the interval of refreshing a Hello packet from an NE to its peer after a Hello session is established. The value ranges from 1s to 25s. By default, the value is 3s.";
        config "true";
        default "3";
        type uint32 {
            range "1..25";
        }
    }
    leaf supportGRENable {
        description "Support GR enabling state.";
        config "true";
        default "false";
        type boolean;
    }
    leaf keepMultiplier {
        description "Timeout multiplier of soft state PSB or RSB. The value ranges from 3 to 255. By default, the value is 3. If the local NE does not receive a packet for refreshing the TE LSP soft state from its upstream or downstream neighbor in the period (that is a specified multiplier of the refreshing cycle), the NE considers that the soft state times out and will delete the TE LSP.";
        config "true";
    }
}

```





```

        default "3";
        type uint32 {
            range "3..255";
        }
    }
    leaf refreshInterval {
        description "Cycle of refreshing a PATH or a RESV message (for s
oft state maintenance). The value ranges from 10 to 65535. By default, the value
is 30.";
        config "true";
        default "30";
        type uint32 {
            range "10..65535";
        }
    }
    leaf srefreshEnable {
        description "Summary refresh enabling state.";
        config "true";
        default "false";
        type boolean;
    }
    leaf retransmissionInterval {
        description "Interval at which interface summary messages are re
transmitted. The value ranges from 500 ms to 5000 ms. By default, the value is 5
000 ms.";
        config "true";
        default "5000";
        type uint32 {
            range "500..5000";
        }
    }
    leaf retransmissionIncrementValue {
        description "Incremental value for retransmitting interface summ
ary. The value ranges from 1 to 10. By default, the value is 1. If an NE does no
t receive a response from the peer end after sending a message to the peer end,
the NE will start the retransmission mechanism. The retransmission interval incr
eases with the increase of the retransmission incremental value. The formula is
as following: retransmission interval = last retransmission interval * (1 + retr
ansmission incremental value).";
        config "true";
        default "1";
        type uint32 {
            range "1..10";
        }
    }
    leaf challengeRetransmissionInterval {
        description "Challenge message retransmission interval. The valu
e ranges from 500 ms to 10000 ms. By default, the value is 1000 ms. If the NE d
oes not receive a response message from the peer end after sending a Challenge m
essage for handshake authentication to the peer end, the NE will retransmit the
Challenge message.";
        config "true";
        default "1000";
        type uint32 {
            range "500..10000";
        }
    }
    leaf maxChallengeMissTimes {
        description "Maximum challenge loss times. The value ranges from
1 to 10. By default, the value is 3. If the times of a retransmitting a Challen
ge message from the NE to its neighbor exceed the maximum challenge loss times,
the NE considers that the handshake cannot be established and will not send a Ch
allenge message again.";

```

```
config "true";  
default "3";  
type uint32 {
```

```

        range "1..10";
    }
}

container rsvpInterfaces {

    list rsvpInterface {

        key "interfaceName";
        max-elements "unbounded";
        min-elements "0";
        description "RSVP interface configuration.";

        leaf interfaceName {
            description "Interface name.";
            config "true";
            type ifName;
        }
        leaf helloEnable {
            description "Hello enabling state.";
            config "true";
            default "false";
            type boolean;
        }
        leaf autoFrrMode {
            description "Auto FRR protection modes: global inheritance,
link protection, node protection and self-adapting. By default, global AutoFRR i
s global inheritance. Link protection: A bypass tunnel can be used to protect on
ly the links where the outbound interface of the primary LSP resides. Node prote
ction: A bypass tunnel can be used to protect the downstream node of the links w
here the outbound interface of the primary LSP resides. Self-adapting??A bypass t
unnel can be dynamically selected node protection or link protection according t
o the network conditions.";
            config "false";
            default "DEFAULT";
            type enumeration {
                enum DEFAULT {
                    value "0";
                    description "DEFAULT:";
                }
                enum LINK {
                    value "1";
                    description "LINK:";
                }
                enum NODE {
                    value "2";
                    description "NODE:";
                }
                enum DISABLE {
                    value "3";
                    description "DISABLE:";
                }
            }
        }
    }
}

```

```

        container authentication {

            description "Neighbor configuration of RSVP TE authentication.";

            leaf authEnable {
                description "Specifies the enabling state of RSVP TE interface authentication.";
                config "true";
                default "false";
                type boolean;
            }
            leaf authMD5Key {
                description "Authentication key of a RSVP TE interface. The key length ranges from 1 to 255 characters. When the key is configured, the RSVP packets that the interface sends out carry the authentication information that are calculated based on the authentication key by using the MD5 algorithm. The packets the interface receives are also verified based on the authentication key. The two ends of a TE link must be configured with the same authentication key, otherwise, RSVP packets cannot pass through the link.";
                config "true";
                type string {
                    length "1..255";
                    pattern "^[^ ]+$";
                }
            }
            leaf authLifetime {
                description "Authentication lifetime. The value ranges from 1 to 86399, in seconds. By default, the value is 1800s. During authentication lifetime, RSVP authentication lifetime will be reset after a RSVP packet is received. If no RSVP packet is received when the authentication lifetime times out, the RSVP neighbor will delete the authentication relationship to prevent persistent authentication.";
                config "true";
                default "1800";
                type uint32 {
                    range "1..86399";
                }
            }
            leaf authHandshake {
                description "Handshake authentication enable. When handshake enabled, the handshake mechanism is configured. When the authentication sequence number of an RSVP packet is out of order, a handshake will be initiated and an authentication will be re-negotiated.";
                config "true";
                type string {
                    length "8..40";
                    pattern "^[^ ]+$";
                }
            }
            leaf authWindowSize {
                description "Size of an authentication window that is set to prevent information loss caused by packet disorder. The value of the size ranges from 1 to 64. By default, the value is 1. If the packet sequence number exceeds the window size, the packet is considered to be out of order.";
                config "true";
                default "1";
                type uint32 {
                    range "1..64";
                }
            }
        }
    }

```

}  
}

```
container cspfCfg {  
    description "CSPF configuration.";   
  
    leaf enableCspf {  
        description "The enabling state of CSPF capability.";   
        config "true";  
        default "false";  
        type boolean;  
    }  
    leaf preferredIgp {  
        description "Configurable entry: preferred Igp";  
        config "true";  
        default "ospf";  
        type enumeration {  
            enum ospf {  
                value "0";  
                description "ospf:";  
            }  
            enum isis {  
                value "1";  
                description "isis:";  
            }  
        }  
    }  
    leaf preferredOspfProcessId {  
        description "preferred ospf process id";  
        config "true";  
        type uint32 {  
            range "1..4294967295";  
        }  
    }  
    leaf preferredOspfAreaId {  
        description "preferred ospf area id";  
        config "true";  
        type boolean;  
    }  
    leaf preferredIsisProcessId {  
        description "preferred isis process id";  
        config "true";  
        type uint32 {  
            range "1..4294967295";  
        }  
    }  
    leaf preferredIsisLevel {  
        description "preferred isis level";  
        config "true";  
        type enumeration {
```

```
        enum invalid {
            value "0";
            description "ISIS invalid";
        }
        enum level1 {
            value "1";
            description "ISIS Level-1";
        }
        enum level2 {
            value "2";
            description "ISIS Level-2";
        }
    }
}
leaf tiebreaking {
    description "Rule of selecting multiple equivalent routes.";
    config "true";
    default "RANDOM";
    type enumeration {
        enum LEASTFILL {
            value "0";
            description "LEASTFILL:The link with the smallest bandwidth occupation ratio is selected in the case of equal conditions.";
        }
        enum MOSTFILL {
            value "1";
            description "MOSTFILL:The link with the largest bandwidth occupation ratio is selected in the case of equal conditions.";
        }
        enum RANDOM {
            value "2";
            description "RANDOM:Links are selected randomly.";
        }
    }
}
leaf pathMetricType {
    description "pre-calc path metric type, TE or IGP.";
    config "true";
    default "TE";
    type enumeration {
        enum IGP {
            value "0";
            description "IGP";
        }
        enum TE {
            value "1";
            description "TE";
        }
    }
}
```



```

    leaf srlgPathCalcMode {
        description "cspf srlg path calculating mode";
        config "true";
        default "default";
        type enumeration {
            enum default {
                value "0";
                description "none: Don't care SRLG when calculate path.";
            }
            enum preferred {
                value "1";
                description "preferred: Try to exclude link in the same
SRLG with excluded path.";
            }
            enum strict {
                value "2";
                description "strict: Never use link in same SRLG with ex
clude path.";
            }
        }
    }
}

container p2mpTeTemplates {

    list p2mpTeTemplate {

        key "templateName";
        max-elements "unbounded";
        min-elements "0";
        description "P2MP TE configuration Template.";

        leaf templateName {
            description "Name of p2mp te configuration template.";
            config "true";
            type string {
                length "0..31";
                pattern "^[^ ]*$";
            }
        }

        leaf recordRouteMode {
            description "Route record mode. The available options are no
t record, recording routes only, and recording routes and labels.";
            config "true";
            default "DISABLE";
            type enumeration {
                enum DISABLE {
                    value "0";
                    description "DISABLE:";
                }
                enum RECORD_ROUTE_ONLY {

```

```

        value "1";
        description "The LSP records routes only.";
    }
    enum RECORD_LABEL {
        value "2";
        description "The LSP records both routes and labels.
";
    }
}
}
leaf resvStyle {
    description "Tunnel reservation styles. SE style: shared explicit style; FF: fixed filter style. The default tunnel reservation style is SE.
";
    config "true";
    default "SE";
    type enumeration {
        enum SE {
            value "0";
            description "Shared Explicit Style:";
        }
        enum FF {
            value "1";
            description "Fixed Filter Style:";
        }
    }
}
leaf setupPriority {
    description "Specifies a tunnel setup priority. The value ranges from 0 to 7. By default, the value is 7. The smaller the value, the higher the setup priority. 0 is the highest priority. Limit: the setup priority of a tunnel must be equal to or smaller than its holding priority.";
    config "true";
    default "7";
    type uint8 {
        range "0..7";
    }
}
leaf holdPriority {
    description "Specifies a tunnel holding priority. The value ranges from 0 to 7. By default, the value is 7. The smaller the value, the higher the holding priority. 0 is the highest priority. Limit: The holding priority of a tunnel must be equal to or larger than its setup priority.";
    config "true";
    default "7";
    type uint8 {
        range "0..7";
    }
}
leaf bandwidth {
    description "Specifies a tunnel bandwidth. The value ranges from 0 kbit/s to 4000000000 kbit/s. By default, the value is 0 kbit/s.";
    config "true";
    default "0";
    type uint32 {
        range "0..4000000000";
    }
}
}

```

```

    leaf reoptimization {
      description "Auto reoptimization enabling state of P2MP Temp
late.";
      config "true";
      default "false";
      type boolean;
    }
    leaf reoptiFrequency {
      description "Frequency of auto reoptimization, its range is
60-604800.";
      config "true";
      default "3600";
      type uint32 {
        range "60..604800";
      }
    }
    leaf pathMetricType {
      description "Referenced metric type of one link for calculat
ing path when creating TE tunnels. The available options are DEFAULT, IGP and TE
, default is inheriting from global configuration.";
      config "true";
      default "NONE";
      type enumeration {
        enum NONE {
          value "0";
          description "Inherit from global configuration.";
        }
        enum IGP {
          value "1";
          description "IGP";
        }
        enum TE {
          value "2";
          description "TE";
        }
      }
    }
    leaf tieBreaking {
      description "Routing rules for a p2mp template with multiple
equal-cost routes. 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 t
he preceding rules.";
      config "true";
      default "DEFAULT";
      type enumeration {
        enum LEASTFILL {
          value "0";
          description "LEASTFILL:The link with the smallest ba
ndwidth occupation ratio is selected in the case of equal conditions.";
        }
        enum MOSTFILL {
          value "1";
          description "MOSTFILL:The link with the largest band
width occupation ratio is selected in the case of equal conditions.";
        }
        enum RANDOM {
          value "2";

```



```
        description "RANDOM:Links are selected randomly.";
    }
    enum DEFAULT {
        value "3";
        description "Inherit from global configuration.";
    }
}
leaf hopLimit {
    description "Number limit of hops in a tunnel path. The value ranges from 1 to 32. By default, the value is 32.";
    config "true";
    default "32";
    type uint32 {
        range "1..32";
    }
}
leaf includeAllAffinity {
    description "Administrative group attribute of an LSP (includeAll).It's hexadecimal string.";
    config "false";
    default "00000000";
    type string {
        length "8";
    }
}
leaf includeAnyAffinity {
    description "Administrative group attribute of an LSP (includeAny).It's hexadecimal string.";
    config "true";
    default "00000000";
    type string {
        length "8";
    }
}
leaf excludeAnyAffinity {
    description "Administrative group attribute of an LSP (excludeAny).It's hexadecimal string.";
    config "true";
    default "00000000";
    type string {
        length "8";
    }
}
leaf leafListName {
    description "Specify the leaf-list.";
    config "true";
    mandatory "true";
    type string {
        length "0..31";
    }
}
```

```

        container mplsTeP2mpTemplateFrr {

            description "Fast reroute attribute.";

            leaf frrEnable {
                description "Request of fast reroute capability.";
                config "true";
                default "false";
                type boolean;
            }
            leaf bwProtEnable {
                description "The tunnel with fast reroute capability req
uests bandwidth protection.";
                config "true";
                default "false";
                type boolean;
            }
            leaf frrBandwidth {
                description "FRR-protection bandwidth (kbits/s) request
ed by an active tunnel. The value ranges from 0 kbit/s to 4000000000 kbit/s. by
default, the value is 0 Kbit/s. The value cannot exceed the bandwidth of the act
ive tunnel.";
                config "true";
                default "0";
                type uint32 {
                    range "0..4000000000";
                }
            }
            leaf frrSetupPriority {
                description "Setup priority of FRR-protection tunnels. T
he value ranges from 0 to 7. By default, the value is 7. The smaller the value,
the higher the setup priority. 0 is the highest priority. Limit: The protection
tunnel setup priority cannot exceed the setup priority of the active tunnel.";
                config "true";
                default "7";
                type uint32 {
                    range "0..7";
                }
            }
            leaf frrHoldPriority {
                description "Holding priority of FRR protection tunnels.
The value ranges from 0 to 7. The smaller the value, the higher the priority. T
he value 0 is the highest priority. Limit: The protection tunnel holding priorit
y cannot exceed the active tunnel holding priority.";
                config "true";
                default "7";
                type uint32 {
                    range "0..7";
                }
            }
        }
    }
}

```

## 5. IANA Considerations

This document makes no request of IANA.

## 6. Security Considerations

This document does not introduce any new security risk.

## 7. Acknowledgements

The authors would like to thank Guangying Zheng, Gang Yan for their contributions to this work.

## 8. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 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.
- [RFC4875] Aggarwal, R., Papadimitriou, D., and S. Yasukawa, "Extensions to Resource Reservation Protocol - Traffic Engineering (RSVP-TE) for Point-to-Multipoint TE Label Switched Paths (LSPs)", RFC 4875, May 2007.
- [RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.
- [RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

## 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

Xinzong Zeng  
Huawei Technologies  
Huawei Bld., No.156 Beiqing Rd.  
Beijing 100095  
China

Email: zengxinzong@huawei.com