TEAS WG
Internet-Draft
Intended status: Standards Track
Expires: 8 January 2026

K. Kompella
V. P. Beeram
C. Ramachandran
Juniper Networks

7 July 2025

RSVP-TE Extensions for Multipath Traffic Engineered Directed Acyclic Graph Tunnels
draft-kbr-teas-mptersvp-01

Abstract

A Multipath Traffic Engineered Directed Acyclic Graph (MPTED) tunnel is a Traffic Engineering (TE) construct that facilitates weighted load balancing of unicast traffic across a constrained set of paths optimized for a specific objective.

This document describes the provisioning of an MPTED Tunnel in a TE network using RSVP-TE.

Status of This Memo

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

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

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

This Internet-Draft will expire on 8 January 2026.

Copyright Notice

Table of Contents

1. Introduction

The notions of Multipath Traffic Engineering (MPTE) and of an MPTE Directed Acyclic Graph (MPTED) tunnel are introduced in [I-D.draft-kompella-teas-mpte]. An MPTED tunnel is a Traffic Engineering (TE) construct that contains a constrained set of paths representing an optimized Directed Acyclic Graph (DAG) from one or more ingresses to one or more egresses. The paths that make up an MPTED tunnel traverse a set of junction nodes, and the state associated with the MPTED at each junction node constitutes a set of previous-hops and a set of next-hops over which traffic is load balanced in a weighted fashion. Provisioning an MPTED tunnel in a TE network using a signaling protocol involves provisioning control and forwarding plane state at each junction node.

As a signaling protocol, RSVP-TE is widely deployed for provisioning point-to-point (P2P) TE tunnels [RFC3209] and point-to-multipoint (P2MP) TE tunnels [RFC4875]. This document discusses the extensions to RSVP-TE for use as a signaling protocol to provision MPTED tunnels. MPTED tunnels provisioned using RSVP-TE are referred to as RSVP MPTED Tunnels.

As discussed in [I-D.draft-kompella-teas-mpte], an MPTED tunnel may be realized over a Multiprotocol Label Switching (MPLS) forwarding plane or a native Internet Protocol (IP) v4/v6 forwarding plane using an appropriate tunnel type. Depending on the deployment needs, a centralized or a distributed approach MAY be adopted for provisioning an MPTED tunnel. The focus of this version of the document is on

discussing how the RSVP-TE protocol is extended to facilitate distributed provisioning of MPTED Tunnels over an MPLS forwarding plane in an intra-domain TE network.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here. These words may also appear in this document in lower case as plain English words, absent their normative meanings.

1.2. Definition of Terms Used in this Document

The reader is expected to be familiar with [I-D.draft-kompella-teas-mpte] where most of the terms used in this document are defined.

1.2.1. Terms Used Specifically in This Document

P2P: point-to-point; (for a tunnel) unicast tunnel with a single ingress and a single egress, typically along a single path

1.3. Comparison with RSVP Multipath Traffic Engineered Container Tunnels

There is a pre-existing (and deployed) attempt to combine TE and multipath using what we can call an "RSVP Multipath Traffic Engineered Container (MPTEC) tunnel". An MPTEC contains multiple dynamically created and individually signaled single-path RSVP P2P tunnels. These member tunnels are dynamically added and removed from the container tunnel at the ingress depending on the amount of traffic steered onto it. Though the container tunnel offers a viable option for facilitating the load balancing of unicast traffic across a constrained set of paths individually optimized for a specific objective, the requirement to individually signal and maintain member LSP state can be a deterrent in specific scaled deployments.

A key differentiator for an MPTED tunnel over an MPTEC tunnel is that with the former, traffic is load-balanced across the next hops at each junction node in the DAG (in a weighted fashion), whereas with the latter, traffic is load-balanced only at the ingress node (and typically equally balanced among the next hops). Another differentiator is that the amount of signaling needed to set up the tunnel is significantly less for the MPTED tunnel compared to the MPTEC tunnel. Finally, a MPTEC tunnel has exactly one ingress and

one egress, whereas an MPTED tunnel can have more than one ingress and/or egress with relatively little extra state; this feature is expected to be popular in BGP and multi-homed VPN deployments.

Consider the reference DAG illustrated in Figure 1. An MPTEC tunnel would need 30 member tunnels to be individually set up to cover all the paths in this DAG, whereas an MPTED would have a single tunnel. Focusing on a single node, with MPTEC, R4 would have 30 PSBs and 30 RSBs, whereas with MPTED, R4 would have a single JSB with 2 previous-hops and 3 next-hops. (The terms PSB, RSB and JSB will be explained in a later section.)

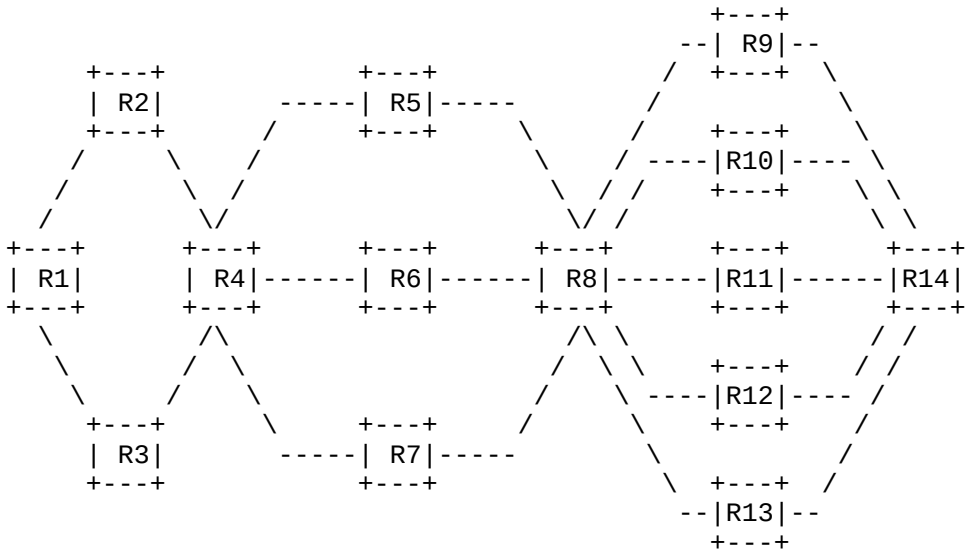


Figure 1: Reference DAG

2. MPTED Tunnels: Overview of Operation

To instantiate an MPTE tunnel in a network via signaling, three steps are needed:

1. Provide the configuration of the MPTED (ingresses, egresses, constraints, etc.) and assign ownership of the DAG to the tunnel originator (TO).

2. Compute an MPTE DAG that satisfies the constraints. This function is undertaken by the MPTE Computer (MC).

3. Signal the required information to the network elements constituting the DAG to establish the tunnel. This task is undertaken by the Signaling Source (SS).

These three functions may be performed by one or more entities. Typical scenarios include:

1. An ingress node of the MPTE DAG does all three functions.

2. An ingress node of the MPTE DAG originates the tunnel, delegates computation of the DAG to a PCE [RFC5440], receives the result and signals the tunnel.

3. A PCE originates the tunnel, computes the DAG and delegates signaling to an ingress node of the DAG.

Other combinations are possible. The subsections that follow describe each function; the next section describes signaling in greater detail.

2.1. MPTED Tunnel Originator

The TO for an MPTED tunnel is typically an ingress of the DAG; however, any node on the DAG can be the TO. In scenarios where the MPTED tunnel has multiple ingress nodes, one of the ingress nodes may be designated as the TO. In deployments where a stateful Path Computation Element (PCE) ([RFC8231], [RFC8281]) model is used to initiate the setup of RSVP MPTED tunnels, the TO is the PCE.

2.1.1. Identification

The TO is responsible for the identity of an MPTED tunnel. An MPTED tunnel is uniquely identified by the 2-tuple: <MPTED Originator ID (MPTED OID), MPTED ID>. The MPTED OID is the IP (v4/v6) (loopback?) address of the TO. An MPTED ID is an unsigned 32-bit positive integer unique to each DAG in the namespace of the MPTED originator (the value 0 is reserved).

2.2. Path Computation

An MPTED may be computed by a path computation engine locally on the TO or by a PCE. In either case, the Traffic Engineering Database (TED) used by the path computation engine is augmented with information indicating whether a topological element supports MPTED tunnel provisioning via RSVP-TE [I-D.kompella-lsr-mptecap]. The path computation request for the MPTED carries an MPTED tunnel ID, a set of ingress nodes, a set of egress nodes, a set of constraints, and an optimization objective. The path computation result for the MPTED

contains a set of unordered elements called JUNCTIONs. This set is communicated to the SS so that the MPTED tunnel can be signaled.

2.2.1. JUNCTION

Each ingress, transit, and egress node on the DAG is a junction and has a JUNCTION element associated with it. A JUNCTION element contains the information necessary to provision a specific junction node in the computed DAG. This version of the document assumes that all junction nodes in the computed DAG are MPTED RSVP capable. The information carried in a JUNCTION element includes the bandwidth coming in and going out of the junction, a list of previous-hops (JCT-PHOPs), and a list of next-hops (JCT-NHOPs).

2.3. MPTED Signaling Source (SS)

The MPTED SS is responsible for creating, maintaining and ultimately destroying an MPTE tunnel. It is handed an MPTED tunnel ID and a set of JUNCTIONs. If signaling is successful, it communicates back to the TO that the tunnel is ready for traffic.

2.3.1. Versioning

The provisioned state associated with the MPTED tunnel may change over time, with each instance of the MPTED tunnel getting assigned an unsigned 32-bit version number (MPTED version). An MPTED tunnel instance is uniquely identified by the 3-tuple <MPTED OID, MPTED ID, MPTED version>. The MPTED version is managed by the SS.

2.3.2. Label Allocation

[I-D.draft-kompella-teas-mpte] offers multiple label allocation schemes for MPTED tunnels realized over an MPLS forwarding plane. Given the presence of a signaling plane, this document advocates the use of the "Signaled Label Switching (SigLab)" approach for RSVP MPTED tunnels.

2.3.3. JUNCTION State Block (JSB)

The control-plane state provisioned on a junction node for a given MPTED Tunnel is referred to as the JUNCTION State Block (JSB). The states pertaining to the JCT-PHOPs and JCT-NHOPs contained in the JSB are referred to as JSB-PHOPs and JSB-NHOPs, respectively.

2.3.4. Tunnel Status

An MPTED tunnel is deemed "Up" if all the junction nodes are provisioned as requested. The tunnel is deemed "Up - Degraded" if some (but not all) paths in the DAG are available for carrying the end-to-end traffic. The tunnel is deemed "Down" if there are no paths in the DAG available for carrying the end-to-end traffic. Based on the difference between the requested bandwidth and the actual reserved bandwidth on the DAG, local policy on the tunnel originator will determine if the MPTED Tunnel should be deemed "Active" (available for traffic to be placed on it) or not.

2.3.5. In-Place Update vs Make-Before-Break

Unless there is a change to the set of constraints used, or an addition or deletion of topological elements, the shape of the computed DAG will remain unchanged over the life of an MPTED tunnel. If the shape of the DAG does not change, the updates to an MPTED tunnel are localized to the bandwidth allotted to the JUNCTION and the relative load shares on the JCT-NHOPs. In such a scenario, the update is carried out in-place and is accompanied by a corresponding version change.

Suppose the shape of the DAG changes for some inevitable reason, meaning there is an addition or deletion of JUNCTIONs or an addition or deletion of JCT-PHOPs/JCT-NHOPs. In that case, the in-place update to the tunnel may cause temporary traffic disruption. Hence, there may be a need to adopt a make-before-break approach to updating the tunnel if the shape of the DAG changes.

3. Signaling for Junction Management

Signaling messages are classified into the following categories:

* (Signaling) Source to Junction node (S2J)

* Junction node to (Signaling) Source (J2S)

* Junction to Junction (J2J)

The underlying RSVP-TE messages used to transmit these messages are analogous to those used in [RFC3209], but are prefixed with M- to distinguish them.

3.1. Source to Junction (S2J) Messages

These are messages signaled from the SS to a junction node on the DAG. The junction node may be an ingress, a transit, or an egress node on the DAG.

3.1.1. JunctionCreate

An S2J JunctionCreate message is used to trigger the instantiation of the "JUNCTION" state on a junction node. Each such message has a version number encoded within it, which identifies the instance of the "JUNCTION" being created.

This document leverages the use of RSVP MPTED Path (M-Path) message to function as an S2J JunctionCreate message.

3.1.2. JunctionUpdate

An S2J JunctionUpdate message is used to trigger the modification of "JUNCTION" state on a junction node. The version number encoded within the message identifies the instance of the "JUNCTION" being modified. The elements of the existing "JUNCTION" entry from the old instance that are no longer part of the DAG are locally tagged as candidates for deletion and remain active until explicitly instructed to do so.

This document leverages the use of RSVP MPTED Path (M-Path) message to function as an S2J JunctionUpdate message.

3.1.3. JunctionDelete

An S2J JunctionDelete message is used to trigger the deletion of the JUNCTION state on a junction node. The message MAY include an instruction to initiate sending a J2J JunctionDelete message to each associated next hop.

This document leverages the use of RSVP MPTED PathTear (M-PathTear) message to function as an S2J JunctionDelete message.

3.2. Junction to Source (J2S) Messages

These are messages signaled from a junction node to the SS.

3.2.1. JunctionNotify

A J2S JunctionNotify message is used to notify the SS of the status of the junction. This message MAY be sent as a response to an S2J message or be sent unsolicited.

This document leverages the use of RSVP MPTED Notify (M-Notify) message to function as a J2S JunctionNotify message.

3.2.2. ResourceNotify

The ResourceNotify message is used to notify the SS of the loss or degradation of an associated resource (e.g., TE link going down, maximum bandwidth on the TE link going down).

This document leverages the use of RSVP ResourceNotify message to function as a J2S ResourceNotify message.

3.3. Junction to Junction (J2J) Messages:

These are messages exchanged between immediately adjacent junction nodes.

3.3.1. Upstream (J2JU) Messages

3.3.1.1. JunctionNextHopReservation

The J2JU JunctionNextHopReserve message is sent to an immediate upstream junction node and is used to facilitate (a) ordered programming of labeled routes at each junction node on the DAG, (b) ordered admission control and bandwidth reservation on traversed TE links, and (c) ordered addition of next hops when changing the shape of the DAG.

This document leverages the use of RSVP MPTED Resv (M-Resv) message to function as a J2JU JunctionNextHopReservation message.

3.3.1.2. JunctionDown

The J2JU JunctionDown message is used to notify an immediate upstream junction node of the local junction state going "Down".

This document leverages the use of RSVP M-Notify message to function as a J2JU JunctionDown message.

3.3.2. Downstream (J2JD) Messages

3.3.2.1. JunctionDelete

The J2JD JunctionDelete message is sent to a JUNCTION next-hop to delete the state, with the condition that the deletion will be propagated further downstream only for next-hops already marked for deletion.

This document leverages the use of RSVP M-PathTear message to function as a J2JD JunctionDelete message.

4. RSVP Messages and Objects

4.1. MPTED Path (M-Path) Message

An M-Path message is an S2J message that is used for creating or updating control and forwarding plane state associated with an MPTED tunnel on a specific junction node. The M-Path message includes the following information:

* MPTED tunnel identifier (SESSION Object)

* MPTED tunnel instance identifier (VERSION Object)

* MPTED tunnel name (SESSION_ATTRIBUTE Object)

* Setup/Hold Priority (SESSION_ATTRIBUTE Object)

* Label type (LABEL_REQUEST Object)

* Junction information - identifier, bandwidth, phops, and nhops with their relative load-shares (<junction-descriptor>)

When a non-egress junction node receives an M-Path message for a new JUNCTION state, it constructs a JSB with the associated JSB-NHOPs and JSB-PHOPs using the information encoded in the message. If the non-egress junction node receives an M-Path for an existing JUNCTION state with a version change, it updates the corresponding JSB using the information encoded in the message. The JSB update may involve adding new JSB-NHOPs and JSB-PHOPs and marking JSB-NHOPs and JSB-PHOPs that are no longer part of the JUNCTION state as candidates for deletion. After the JSB is constructed or updated, the non-egress junction node waits for an M-Resv message to be received from each available JCT-NHOP.

When an egress junction node receives an M-Path message for a new JUNCTION state, it constructs a JSB, assigns a label for each JCT-PHOP, and programs the forwarding plane state, thus completing the JUNCTION provisioning at the egress. If the egress junction node receives an M-Path message for an existing JUNCTION state, it updates the corresponding JSB using the information encoded in the message. The JSB update may involve adding new JSB-PHOPs, and marking JSB-PHOPs that are no longer part of the JUNCTION state as candidates for deletion. After the JSB is constructed/updated, the egress junction node sends an MPTED Resv (M-Resv) message to each JCT-PHOP, and an MPTED Notify (M-Notify) message directly to the tunnel signaling source.

```
<M-Path Message> ::= <Common Header> [<INTEGRITY>]
                     [ [<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ... ]
                     [ <MESSAGE_ID> ]
                     <SESSION> [<END_POINTS>]
                     <TIME_VALUES> <VERSION>
                     <LABEL_REQUEST> <SESSION_ATTRIBUTE>
                     <junction-descriptor>

<junction-descriptor> ::= <JUNCTION> <junction-elements>
<junction-elements> ::=  ( <JUNCTION_PHOPS> | <JUNCTION_NHOPS> |
                           ( <JUNCTION_PHOPS> <JUNCTION_NHOPS> ))
```

4.2. MPTED Resv (M-Resv) Message

An M-Resv message is a J2J message that is used to signal the label that an upstream junction node needs to program for a specific next hop. The M-Resv message includes the following information:

* MPTED tunnel identifier (SESSION Object)

* MPTED tunnel instance identifier (VERSION Object)

* Hop specific information - Hop identifier, Label, and MTU (a list of JUNCTION_LABELED_HOP Objects)

When a transit junction node receives an M-Resv message from all available JCT-NHOPs, it performs admission control, assigns a label to each JCT-PHOP, programs the forwarding plane state, and sends an M-Resv message to each JCT-PHOP and an M-Notify message directly to the tunnel signaling source. No message is sent out until M-Resv messages from all available JCT-NHOPs have been received and processed.

When an ingress junction node receives an M-Resv message from all available JCT-NHOPs, it performs admission control, programs the forwarding plane state, and notifies the tunnel signaling source.

```
<M-Resv Message> ::= <Common Header> [ <INTEGRITY> ]
                     [ [<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ... ]
                     [ <MESSAGE_ID> ]
                     <SESSION>
                     <TIME_VALUES> <VERSION>
                     <junction-labeled-hops-list>

<junction-labeled-hops-list> ::= <JUNCTION_LABELED_HOP>
                                 [ <junction-labeled-hops-list> ]
```

4.3. MPTED Pathtear (M-PathTear) Message

An M-PathTear message may be used as either an S2J message or a J2J message. When an S2J M-PathTear is used for deleting the state on a junction node, the message includes the following information:

* MPTED tunnel identifier (SESSION Object)

* MPTED tunnel instance identifier (VERSION Object)

* Optionally, an instruction to propagate the deletion request downstream (CONDITIONS Object)

When a junction node receives an S2J M-PathTear message, it deletes the matching JSB. It sends an M-Notify message to the tunnel signaling source, indicating that the junction deletion is complete. If the M-PathTear carries an optional instruction to propagate the deletion further downstream, the junction node sends a J2J M-PathTear to each associated JCT-NHOP before deleting the JSB.

When a J2J M-Pathtear is used for deleting a specific hop state on a downstream junction node, the message includes the following information:

* MPTED tunnel identifier (SESSION Object)

* MPTED tunnel instance identifier (VERSION Object)

* Hop identifier (JUNCTION_HOP Object)

During the make-before-break update of an MPTED tunnel, when a junction node completes updating all JCT-PHOPs matching the new version, and determines that there are no JCT-PHOPs pending deletion, it checks if there are any JCT-NHOPs marked for deletion. If such JCT-NHOPs exist, the junction node sends a J2J M-PathTear for each of those JCT-NHOPs with the old version.

When a junction node receives a J2J M-PathTear, it cleans up the corresponding JCT-PHOP state. If there are no other JCT-PHOPs, then it cleans up the JSB and propagates the J2J M-PathTear to each associated JCT-NHOP. If there are other JCT-PHOPs present, but none of them are pending deletion, then it propagates the J2J M-PathTear only to those JCT-NHOPs that have already been marked for deletion.

```
<M-PathTear Message> ::= <Common Header> [ <INTEGRITY> ]
                         [ [<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ... ]
                         [ <MESSAGE_ID> ]
                         <SESSION> <VERSION>
                         [ <JUNCTION_HOP> ] | [ <CONDITIONS> ]
```

4.4. MPTED Notify (M-Notify) Message

An M-Notify message may be used as either a J2S message or a J2J message. A junction node sends a J2S M-Notify message to the tunnel signaling source to indicate the status of the junction. A junction node may send a J2S M-Notify message in response to an S2J message or unsolicited. A J2S M-Notify message includes the following information:

* MPTED tunnel identifier (SESSION Object)

* MPTED tunnel instance identifier (VERSION Object)

* MTU (JUNCTION_STATUS Object)

* Status (JUNCTION_STATUS Object)

If the Status is not "Degraded", the M-Notify message includes the following additional information:

* Reserved bandwidth on the junction (JUNCTION_STATUS Object)

* List of JCT-PHOPs that are "Down"

* List of JCT-NHOPs that are "Down/Degraded" and the reserved bandwidth on each corresponding TE link.

A junction node sends a J2J M-Notify message to the upstream junction node to indicate that it is "Down" . A J2J M-Notify message includes the following information:

* MPTED tunnel identifier (SESSION Object)

* MPTED tunnel instance identifier (VERSION Object)

* Hop Identifier (JUNCTION_HOP_STATUS Object)

* Status (JUNCTION_HOP_STATUS Object)

When an upstream junction node receives a J2J M-Notify indicating that the junction on the specified JCT-NHOP is "Down", it sets the load-share on the JCT-NHOP to "zero" and reprograms the labeled routes.

```
<M-Notify Message> ::= <Common Header> [<INTEGRITY>]
                       [ [<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ... ]
                       [ <MESSAGE_ID> ]
                       <SESSION> <VERSION>
                       (<JUNCTION_HOP_STATUS> |
                        <junction status descriptor>)
<junction status descriptor> ::= <JUNCTION_STATUS>
                                 [ <degraded junction-elements> ]
<degraded junction-elements> ::= ( <JUNCTION_PHOPS> |
                                   <JUNCTION_NHOPS> |
                                   ( <JUNCTION_PHOPS>
                                     <JUNCTION_NHOPS> ))
```

4.5. Resource Notify (RsrcNotify) Message

A RsrcNotify is a J2S message that is used to notify the tunnel signaling source of link unavailability or degradation. A RsrcNotify message includes the following information:

* A list of unavailable resources (list of RESOURCE_SPEC objects)

* A list of degraded resources (list of DEG_RESOURCE_SPEC objects).

When a TE link goes down, the junction node sends a RsrcNotify to notify each impacted tunnel signaling source that the specified TE link is no longer available.

When the maximum reservable bandwidth of a TE link is reduced (for example, a member link on an Aggregate Ethernet link fails), the junction node selects a set of impacted tunnel signaling sources and notifies them that the specified TE link has diminished capacity. In

this scenario, the information carried in the RsrcNotify message is customized for the recipient. It includes the amount of per-priority bandwidth usage that the tunnel signaling source would need to reduce on that TE link.

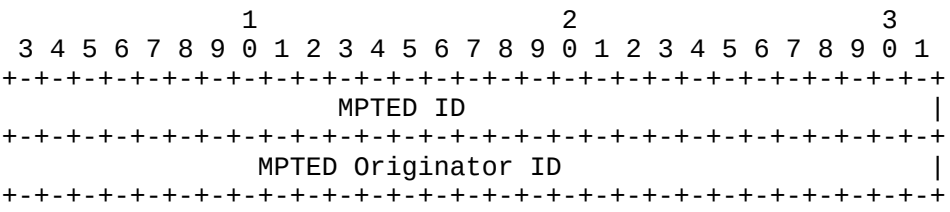
```
<ResourceNotify Message> ::= <Common Header> [<INTEGRITY>]
                      [ [<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ... ]
                      [ <MESSAGE_ID> ]
                      ( <unavailable-resources> |
                        <degraded-resources> |
                        ( <unavailable-resources>
                          <degraded-resources> ))

<unavailable resources> ::=
       (( <RESOURCE_SPEC> <unavailable resources> ) |
        <RESOURCE_SPEC> )
<degraded resources> ::=
       (( <DEG_RESOURCE_SPEC> <degraded resources> ) |
        <DEG_RESOURCE_SPEC> )
```

4.6. Objects

4.6.1. SESSION Object

4.6.1.1. MPTED IPv4 Session

Class = SESSION, MPTED_IPv4 C-Type = TBD



* MPTED ID:

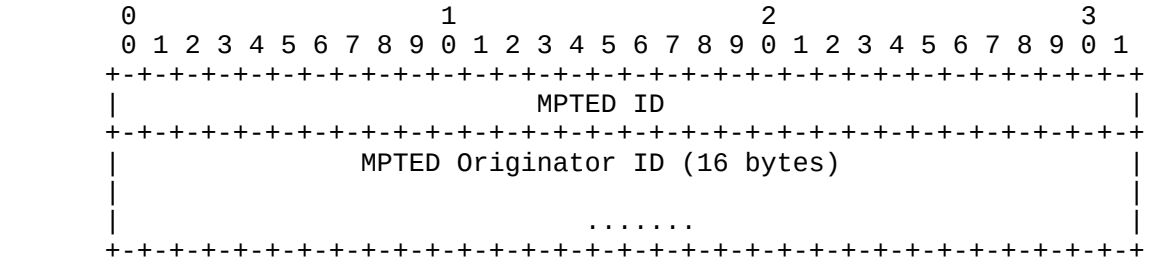
  - A 32 bit identifier that remains constant over the life of the MPTED tunnel.

* MPTED Originator ID:

  - IPv4 address of the originator of the MPTED tunnel.

4.6.1.2. MPTED IPv6 Session

Class = SESSION, MPTED_IPv6 C-Type = TBD



* MPTED ID:

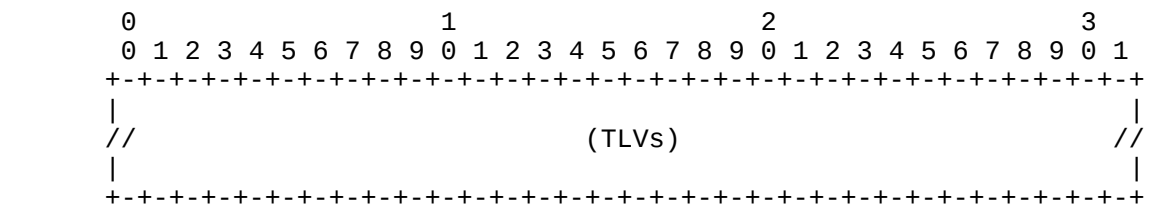
  - A 32 bit identifier that remains constant over the life of the MPTED tunnel.

* MPTED Originator ID:

  - IPv6 address of the originator of the MPTED tunnel.

4.6.2. END POINTS Object

Class = END_POINTS (TBD), C-Type = TBD

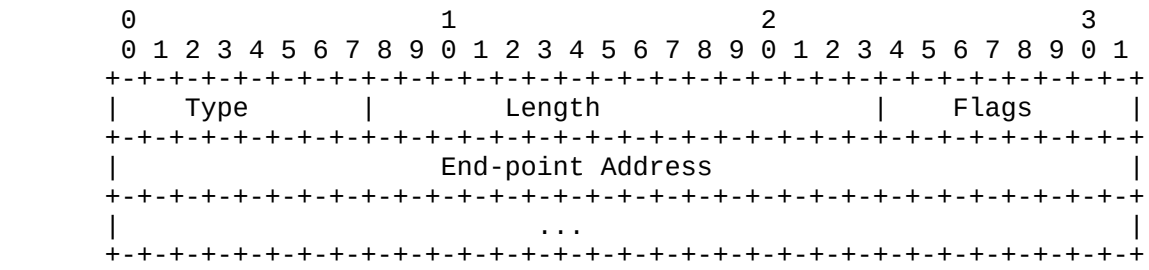


* TLVs

  - The contents of an END_POINTS object are a series of TLVs.

If an M-Path message contains multiple END_POINTS objects, only the first object is meaningful. Subsequent END_POINTS objects MAY be ignored and SHOULD NOT be propagated.

4.6.2.1. IPv4_Endpoints TLV



* Type:

  - 0x01 Series of IPv4 Endpoints.

* Length:

  - Total length of the TLV. It varies based on the number of addresses encoded.

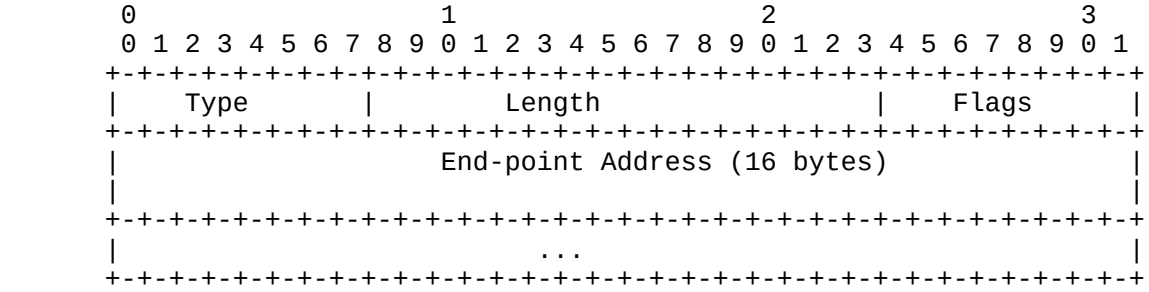
* Flags:

  - 0x01 Ingress (I): Presence of this flag denotes all specified addresses in the TLV as ingresses and its absence denotes all addresses in the TLV as egresses.

* End-point Address

  - Ipv4 address of the endpoint.

4.6.2.2. IPv6_Endpoints TLV



* Type:

  - 0x02 Series of IPv6 Endpoints.

* Length:

- Total length of the TLV. It varies based on the number of addresses encoded.

* Flags:

- 0x01 Ingress (I): Presence of this flag denotes all specified addresses in the TLV as ingresses and its absence denotes all addresses in the TLV as egresses.

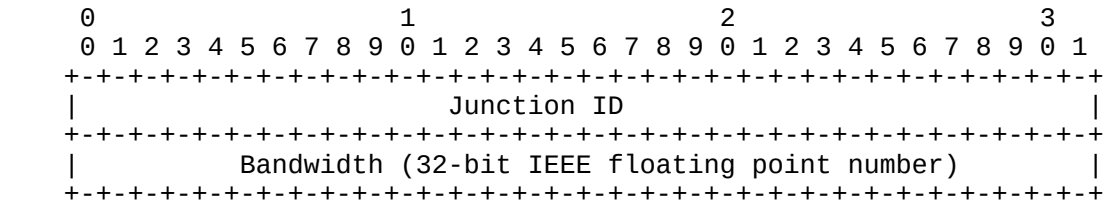
* End-point Address

- Ipv6 address of the endpoint.

4.6.3. JUNCTION Object

4.6.3.1. JUNCTION_IPv4

Class = JUNCTION, JUNCTION_IPv4 C-Type = TBD



* Junction ID:

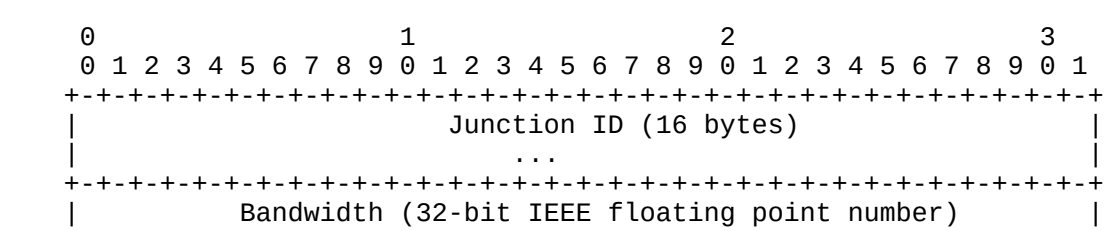
- IPv4 address of the junction node.

* Bandwidth:

- Bandwidth coming into the junction node (encoded as a 32-bit IEEE floating point number). The units are bytes per second.

4.6.3.2. JUNCTION_IPv6

Class = JUNCTION, JUNCTION_IPv6 C-Type = TBD



* Junction ID:

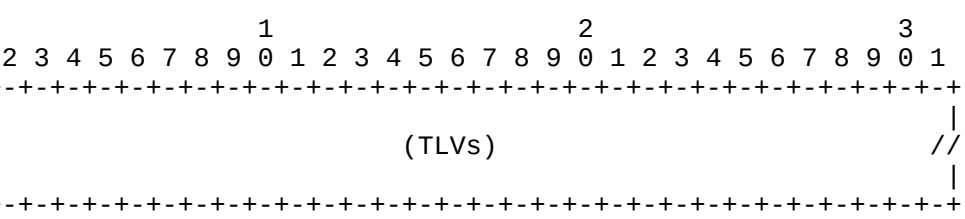
  - IPv6 address of the junction node.

* Bandwidth:

  - Bandwidth coming into the junction node (encoded as a 32-bit IEEE floating point number). The units are bytes per second.

4.6.4. JUNCTION PHOPS Object

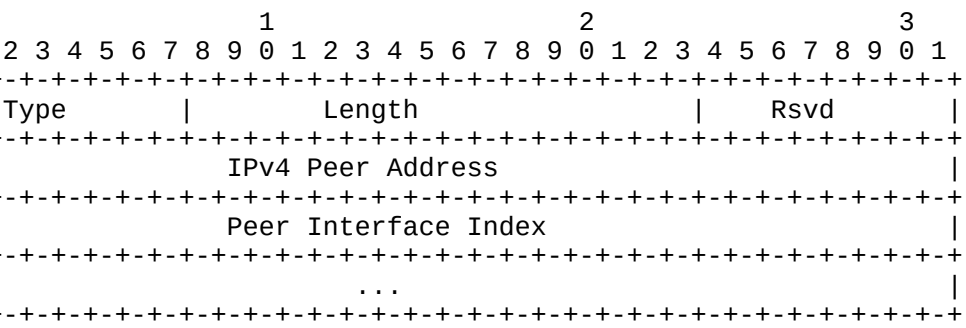
Class = JUNCTION_PHOPS (TBD), C-Type = TBD



* TLVs:

  - The contents of a JUNCTION_PHOPS object are a series of TLVs.

4.6.4.1. JUNCTION_PHOP_IPv4 TLV



* Type:

  - 0x01 Series of IPv4 PHOPs

* Length:

  - Total length of the TLV. It varies based on the number of IPv4 PHOPs encoded.

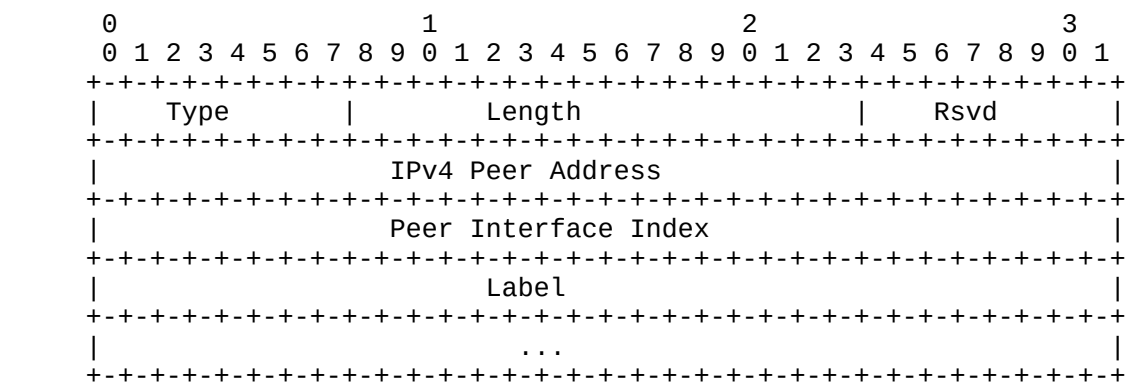
* IPv4 Peer Address:

  - IPv4 address of the previous hop (remote end of the TE link)

* Peer Interface Index:

  - Index of the peer interface (remote end of the TE link)

4.6.4.2. JUNCTION_PHOP_IPv4_Label TLV



* Type:

  - 0x02 Series of IPv4 PHOPs with respective assigned labels.

* Length:

  - Total length of the TLV. It varies based on the number of entities encoded.

* IPv4 Peer Address:

  - IPv4 address of the previous hop (remote end of the TE link).

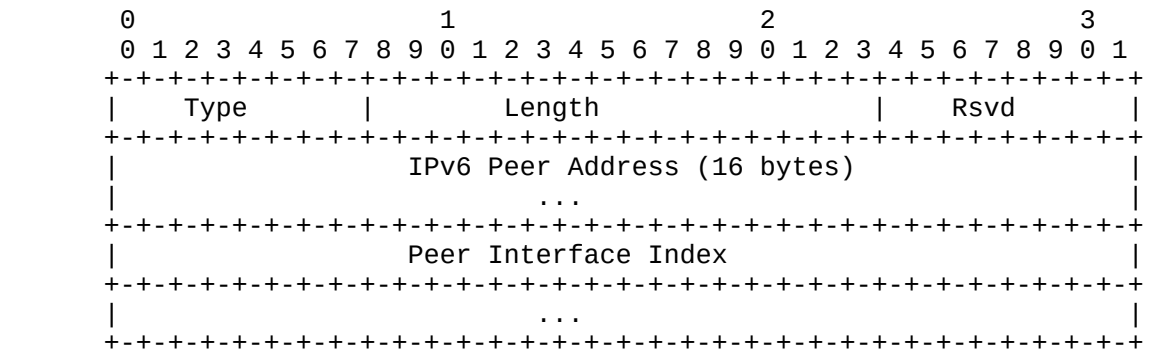
* Peer Interface Index:

  - Index of the peer interface (remote end of the TE link).

* Label:

  - Assigned label for the PHOP.

4.6.4.3. JUNCTION_PHOP_IPv6 TLV



* Type:

  - 0x03 Series of IPv6 PHOPs

* Length:

  - Total length of the TLV. It varies based on the number of IPv6 PHOPs encoded.

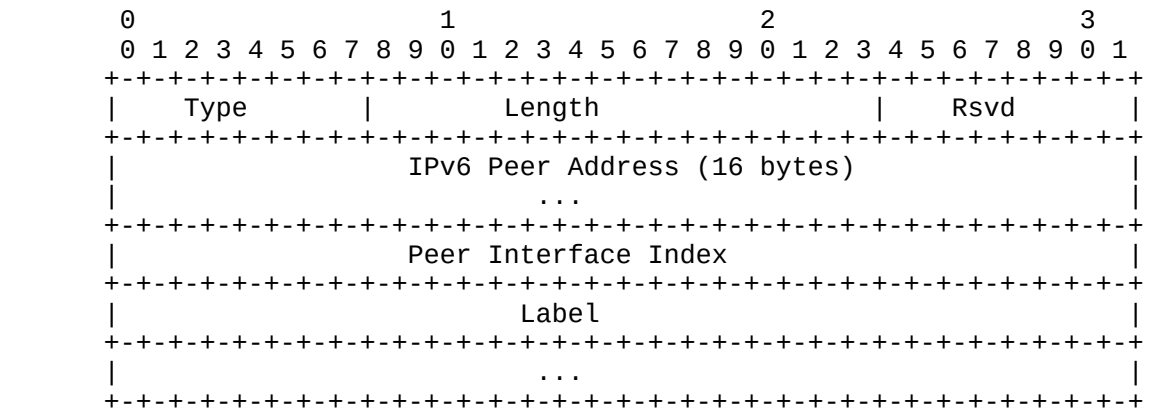
* IPv6 Peer Address:

  - IPv6 address of the previous hop (remote end of the TE link).

* Peer Interface Index:

  - Index of the peer interface (remote end of the TE link).

4.6.4.4. JUNCTION_PHOP_IPv6_Label TLV



* Type:

- 0x04 Series of IPv6 PHOPs with respective assigned labels.

* Length:

  - Total length of the TLV. It varies based on the number of entities encoded.

* IPv6 Peer Address:

  - IPv6 address of the previous hop (remote end of the TE link).

* Peer Interface Index:

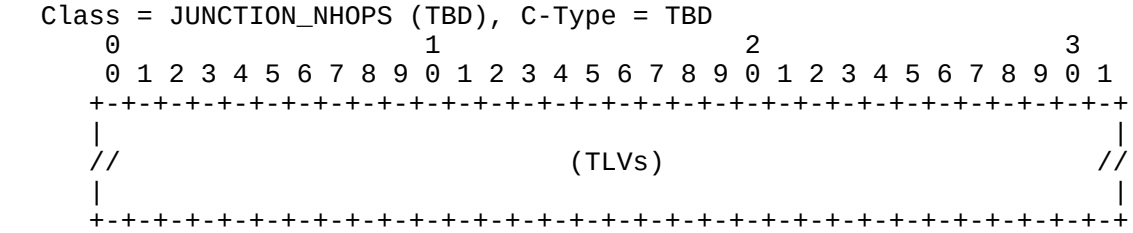
  - Index of the peer interface (remote end of the TE link).

* Label:

  - Assigned label for the PHOP.

4.6.5. JUNCTION NHOPS Object

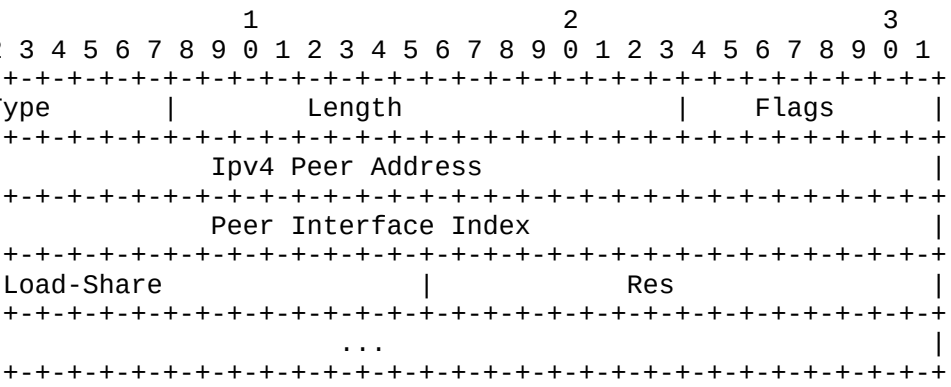
Class = JUNCTION_NHOPS (TBD), C-Type = TBD



* TLVs:

  - The contents of a JUNCTION_NHOPS object are a series of TLVs.

4.6.5.1. JUNCTION_NHOP_IPv4 TLV



* Type:

  - 0x01 Series of IPv4 NHOPs

* Length:

  - Total length of the TLV. It varies based on the number of IPv4 NHOPs encoded.

* Flags:

  - 0x01 Backup (B): Presence of this flag denotes the specified next-hop as a backup next-hop.

* IPv4 Peer Address:

  - IPv4 address of the previous hop (remote end of the TE link).

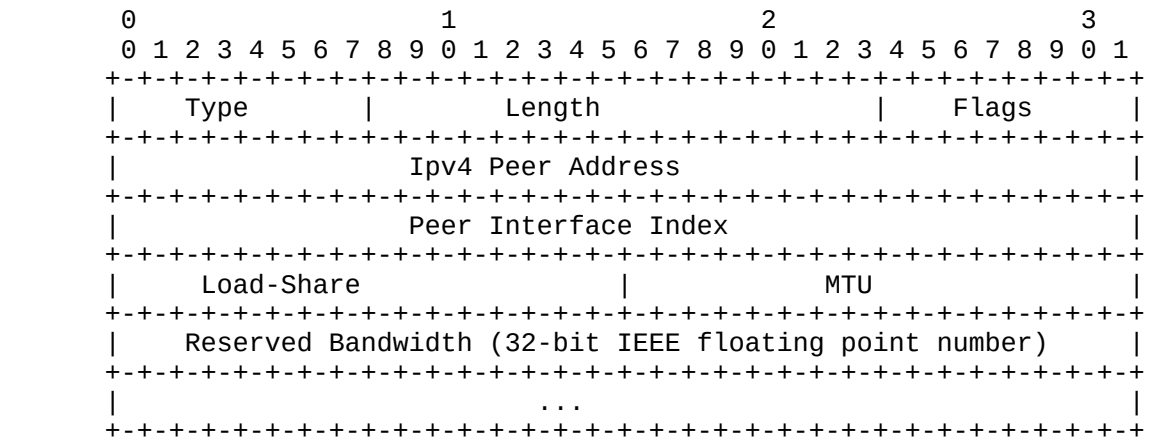
* Peer Interface Index:

  - Index of the peer interface (remote end of the TE link).

* Load-Share:

  - Relative share of the outgoing bandwidth.

4.6.5.2. JUNCTION_NHOP_IPv4_STATUS TLV



* Type:

  - 0x02 Series of IPv4 NHOP Statuses

* Length:

  - Total length of the TLV. It varies based on the number of IPv4 NHOP statuses encoded.

* Flags:

  - 0x01 Backup (B): Presence of this flag denotes the specified next-hop as a backup next-hop.

* IPv4 Peer Address:

  - IPv4 address of the previous hop (remote end of the TE link).

* Peer Interface Index:

  - Index of the peer interface (remote end of the TE link).

* Load-Share:

  - Relative share of the outgoing bandwidth.

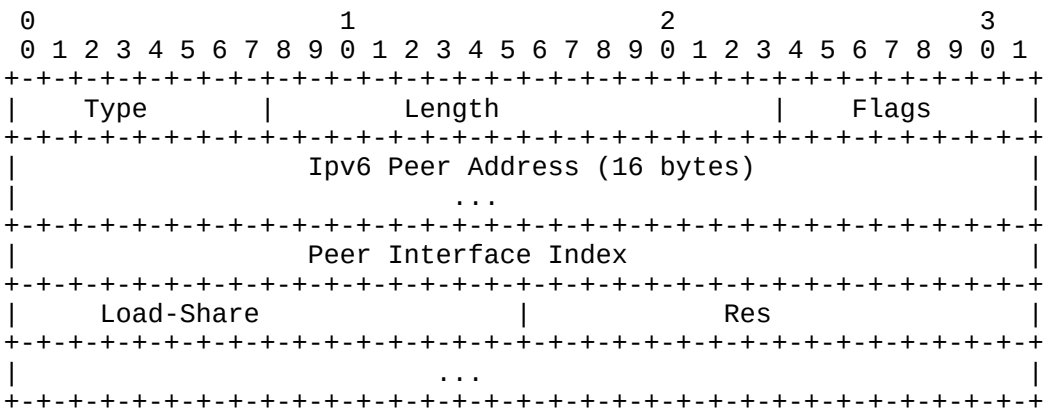
* MTU:

  - MTU value received from the next-hop

* Reserved Bandwidth:

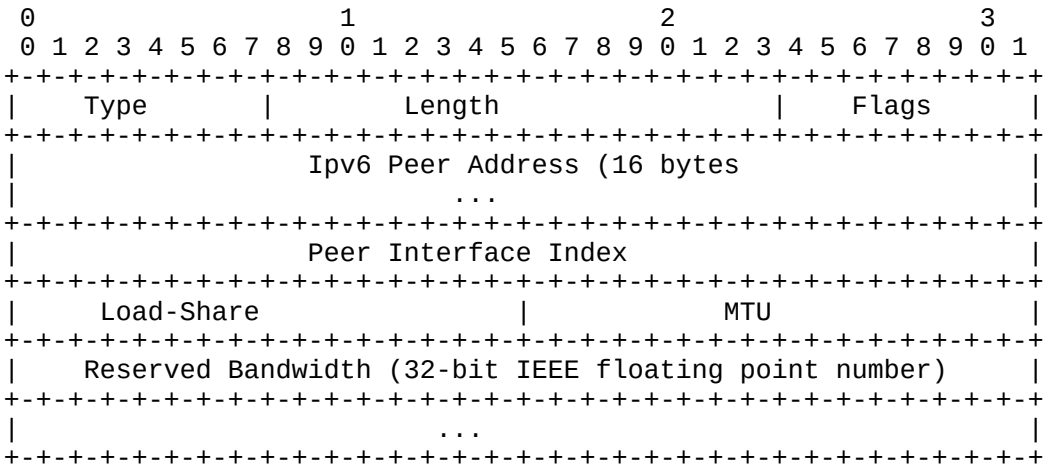
- Bandwidth reserved on the TE link associated with the next-hop (encoded as a 32-bit IEEE floating point number). The units are bytes per second.

4.6.5.3. JUNCTION_NHOP_IPv6 TLV



* Type:
  - 0x03 Series of IPv6 NHOPs
* Length:
  - Total length of the TLV. It varies based on the number of IPv6 NHOPs encoded.
* Flags:
  - 0x01 Backup (B): Presence of this flag denotes the specified next-hop as a backup next-hop.
* IPv6 Peer Address:
  - IPv6 address of the previous hop (remote end of the TE link).
* Peer Interface Index:
  - Index of the peer interface (remote end of the TE link).
* Load-Share:
  - Relative share of the outgoing bandwidth.

4.6.5.4. JUNCTION_NHOP_IPv6_STATUS TLV



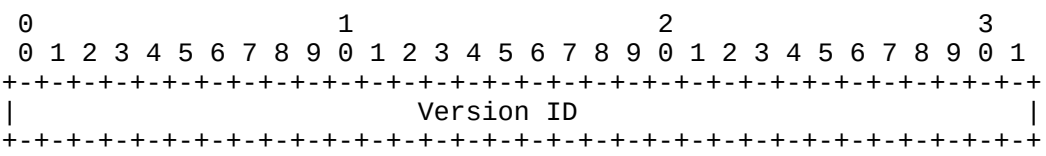
- Type:
  - 0x04 Series of IPv6 NHOP Statuses
- Length:
  - Total length of the TLV. It varies based on the number of IPv6 NHOP statuses encoded.
- Flags:
  - 0x01 Backup (B): Presence of this flag denotes the specified next-hop as a backup next-hop.
- IPv6 Peer Address:
  - IPv6 address of the previous hop (remote end of the TE link).
- Peer Interface Index:
  - Index of the peer interface (remote end of the TE link).
- Load-Share:
  - Relative share of the outgoing bandwidth.
- MTU:
  - MTU value received from the next-hop

* Reserved Bandwidth

  - Bandwidth reserved on the TE link associated with the next-hop (encoded as a 32-bit IEEE floating point number). The units are bytes per second.

4.6.6. VERSION Object

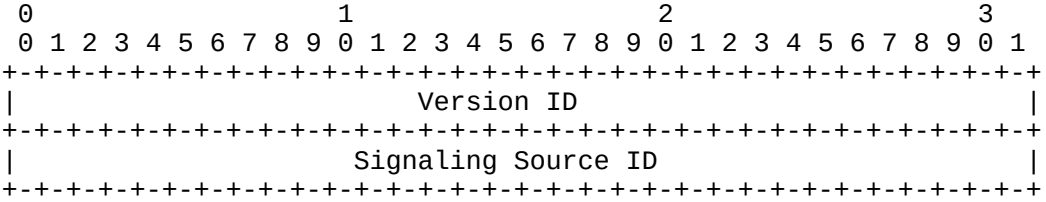
Class = VERSION, Version C-Type = TBD



Class = VERSION, Version_SS_v4 C-Type = TBD

* Version ID:

  - Instance identifier of the MPTED tunnel.



* Version ID:

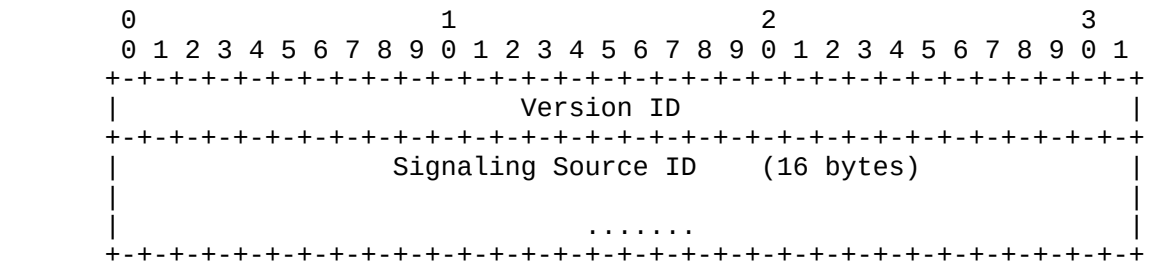
  - Instance identifier of the MPTED tunnel.

* Signaling Source ID:

  - IPv4 address of the MPTED tunnel signaling source.

This C-Type is used only if the MPTED tunnel originator is different from the signaling source.

Class = VERSION, Version_SS_v6 C-Type = TBD



* Version ID:

  - Instance identifier of the MPTED tunnel.

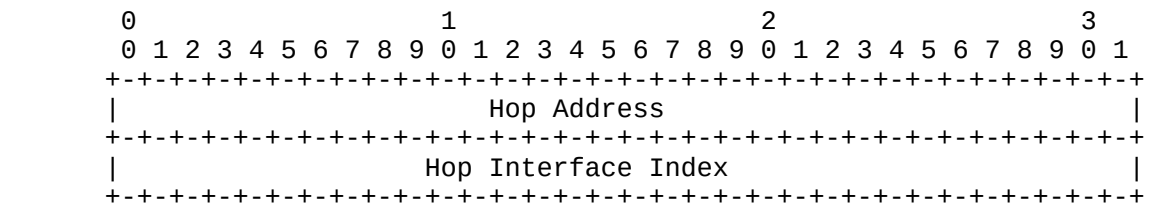
* Signaling Source ID:

  - IPv6 address of the MPTED tunnel signaling source.

This C-Type is used only if the MPTED tunnel originator is different from the signaling source.

4.6.7. JUNCTION HOP Object

Class = JUNCTION_HOP, JUNCTION_HOP_v4 C-Type = TBD



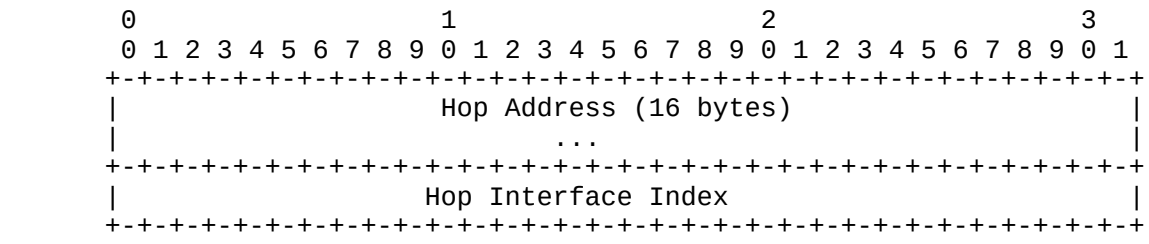
* Hop Address:

  - IPv4 address of the hop of interest.

* Hop Interface Index:

  - Interface index of the hop of interest.

Class = JUNCTION_HOP, JUNCTION_HOP_v6 C-Type = TBD



* Hop Address:

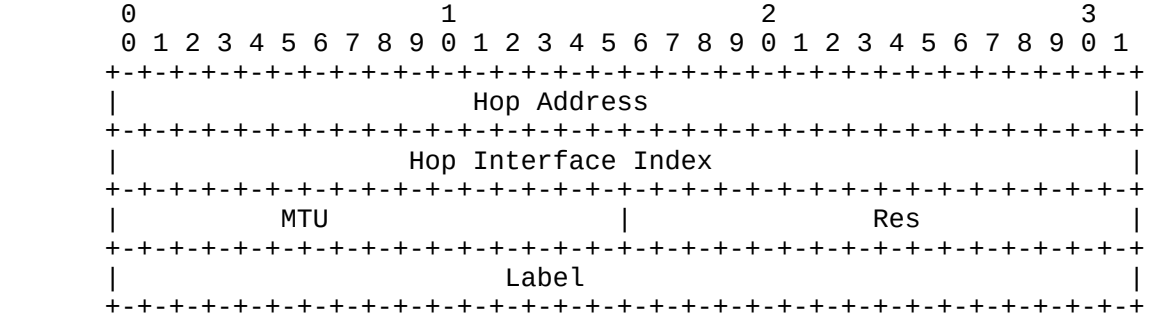
  - IPv6 address of the hop of interest.

* Hop Interface Index:

  - Interface index of the hop of interest.

4.6.8. JUNCTION_LABELED_HOP Object

Class = JUNCTION_LABELED_HOP, JUNCTION_LABELED_HOP_IPv4_Label C-Type = TBD



* Hop Address:

  - IPv4 address of the hop of interest.

* Hop Interface Index:

  - Interface index of the hop of interest.

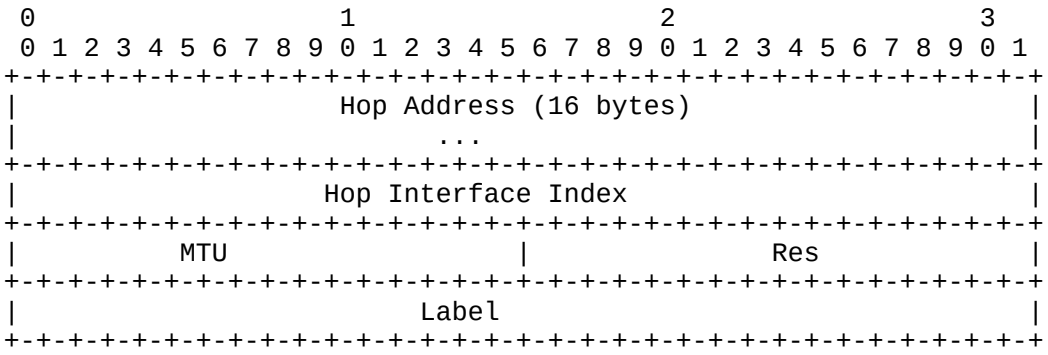
* MTU:

  - MTU value associated with the specified hop.

* Label:

- Label associated with the specified hop.

Class = JUNCTION_LABELED_HOP, JUNCTION_LABELED_HOP_IPv6 TLV C-Type = TBD



* Hop Address:

  - IPv6 address of the hop of interest.

* Hop Interface Index:

  - Interface index of the hop of interest.

* MTU:

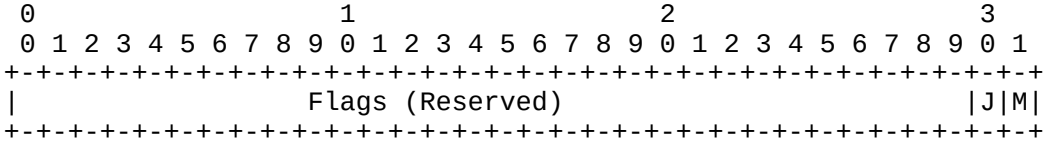
  - MTU value associated with the specified hop.

* Label:

  - Label associated with the specified hop.

4.6.9. CONDITIONS Object

Class = CONDITIONS, C-Type = 1

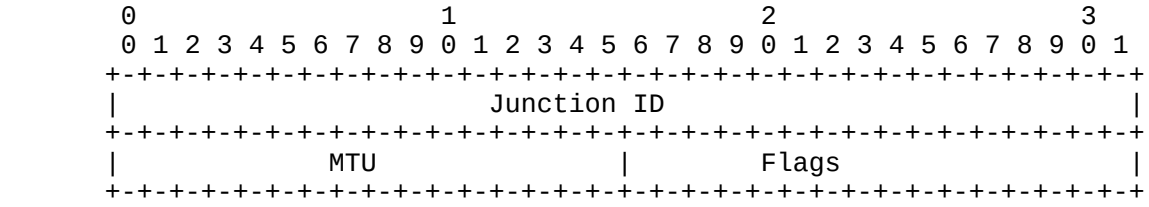


* Flags:

  - J-Bit: Propagate the deletion request downstream.

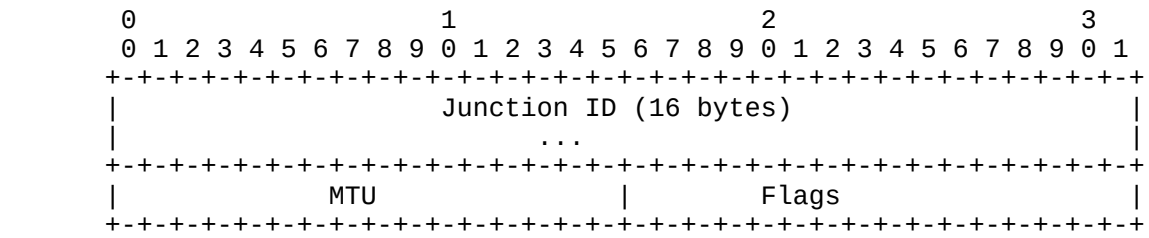
4.6.10. JUNCTION_STATUS

Class = JUNCTION_STATUS, JUNCTION_STATUS_Ipv4 C-Type = TBD



* Junction ID:
  - IPv4 address of the junction node.
* MTU:
  - MTU of the junction node.
* Flags:
  - U-Bit: UP Status.
  - D-Bit: Down Status.
  - G-Bit: Degraded Status.

Class = JUNCTION_STATUS, JUNCTION_STATUS_Ipv6 C-Type = TBD



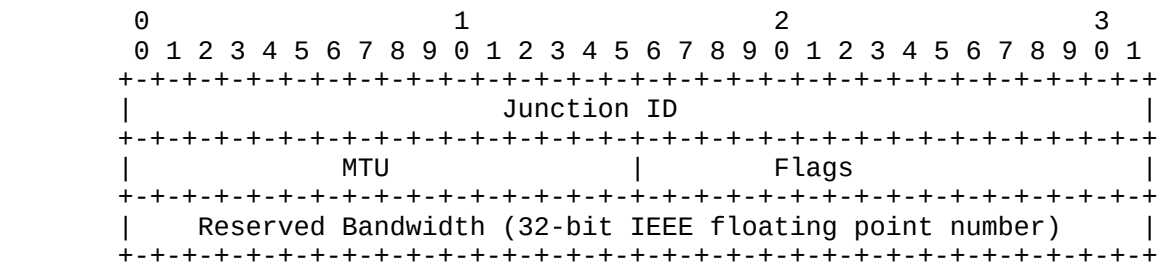
* Junction ID:
  - IPv6 address of the junction node.
* MTU:
  - MTU of the junction node.
* Flags:

- U-Bit: UP Status.

- D-Bit: Down Status.

- G-Bit: Degraded Status.

Class = JUNCTION_STATUS, JUNCTION_STATUS_Degraded_Ipv4 C-Type = TBD



* Junction ID:

  - IPv4 address of the junction node.

* MTU:

  - MTU of the junction node.

* Flags:

  - U-Bit: UP Status.

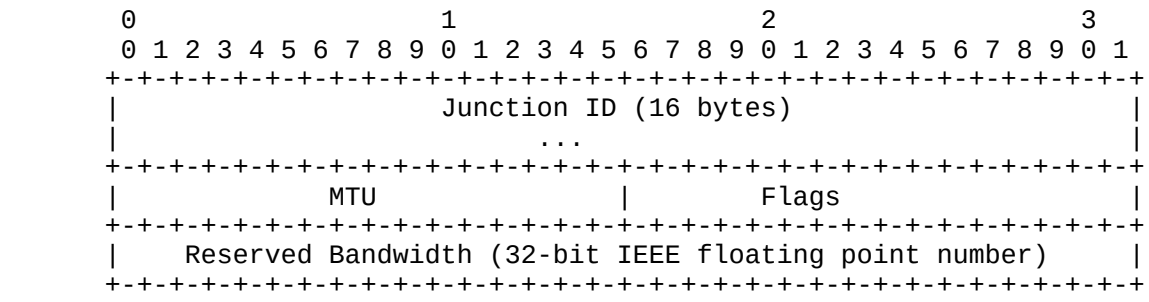
  - D-Bit: Down Status.

  - G-Bit: Degraded Status.

* Reserved Bandwidth:

  - Bandwidth reserved for the junction (encoded as a 32-bit IEEE floating point number).

Class = JUNCTION_STATUS, JUNCTION_STATUS_Degraded_Ipv6 C-Type = TBD



* Junction ID:

  - IPv6 address of the junction node.

* MTU:

  - MTU of the junction node.

* Flags:

  - U-Bit: UP Status.

  - D-Bit: Down Status.

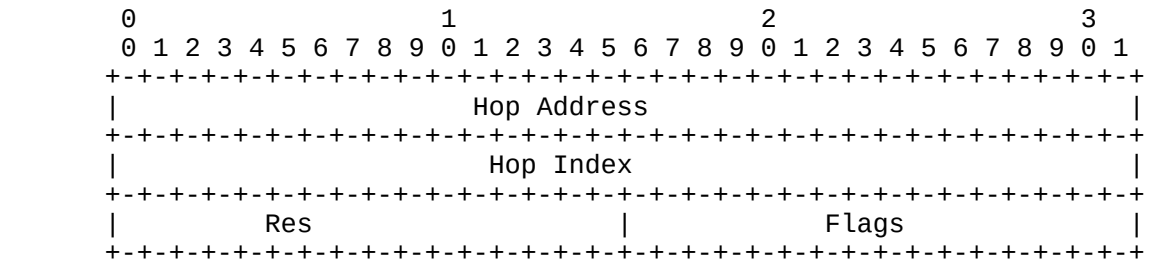
  - G-Bit: Degraded Status.

* Reserved Bandwidth:

  - Bandwidth reserved for the junction (encoded as a 32-bit IEEE floating point number).

4.6.11. JUNCTION_HOP_STATUS

Class = JUNCTION_HOP_STATUS, JUNCTION_HOP_IPv4_STATUS C-Type = TBD



* Hop Address:

- IPv4 address of the hop of interest.

* Hop Index:

- Interface index of the hop of interest.

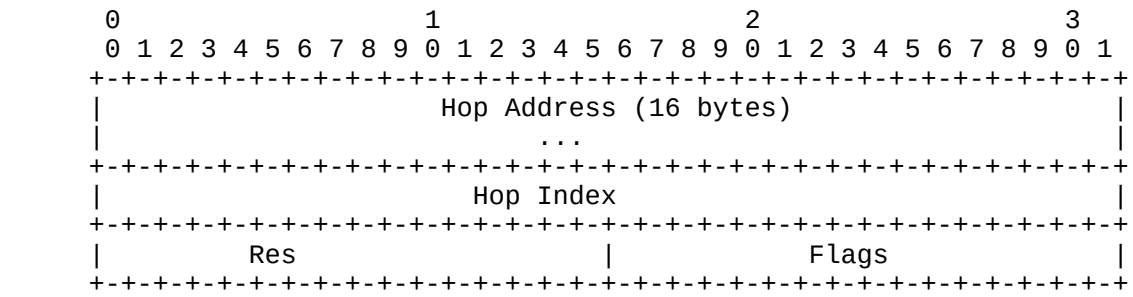
* Flags:

- U-Bit: UP Status.

- D-Bit: Down Status.

- G-Bit: Degraded Status.

Class = JUNCTION_HOP_STATUS, JUNCTION_HOP_IPv6_STATUS C-Type = TBD



* Hop Address:

- IPv6 address of the hop of interest.

* Hop Index:

- Interface index of the hop of interest.

* Flags:

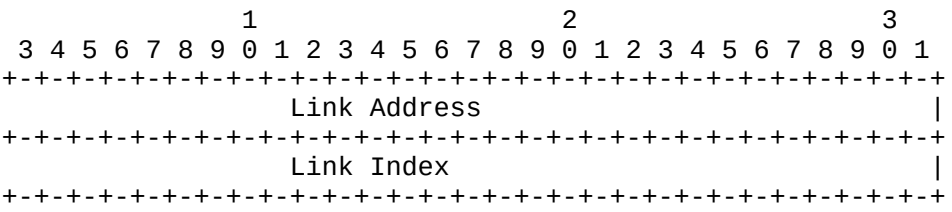
- U-Bit: UP Status.

- D-Bit: Down Status.

- G-Bit: Degraded Status.

4.6.12. RESOURCE_SPEC

Class = RESOURCE_SPEC, RESOURCE_SPEC_Ipv4 C-Type = TBD



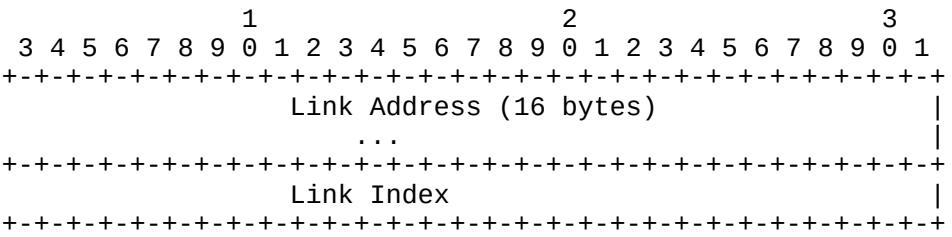
* Link Address:

  - IPv4 address of the unavailable TE link.

* Link Index:

  - Index of the unavailable TE link.

Class = RESOURCE_SPEC, RESOURCE_SPEC_Ipv6 C-Type = TBD



* Link Address:

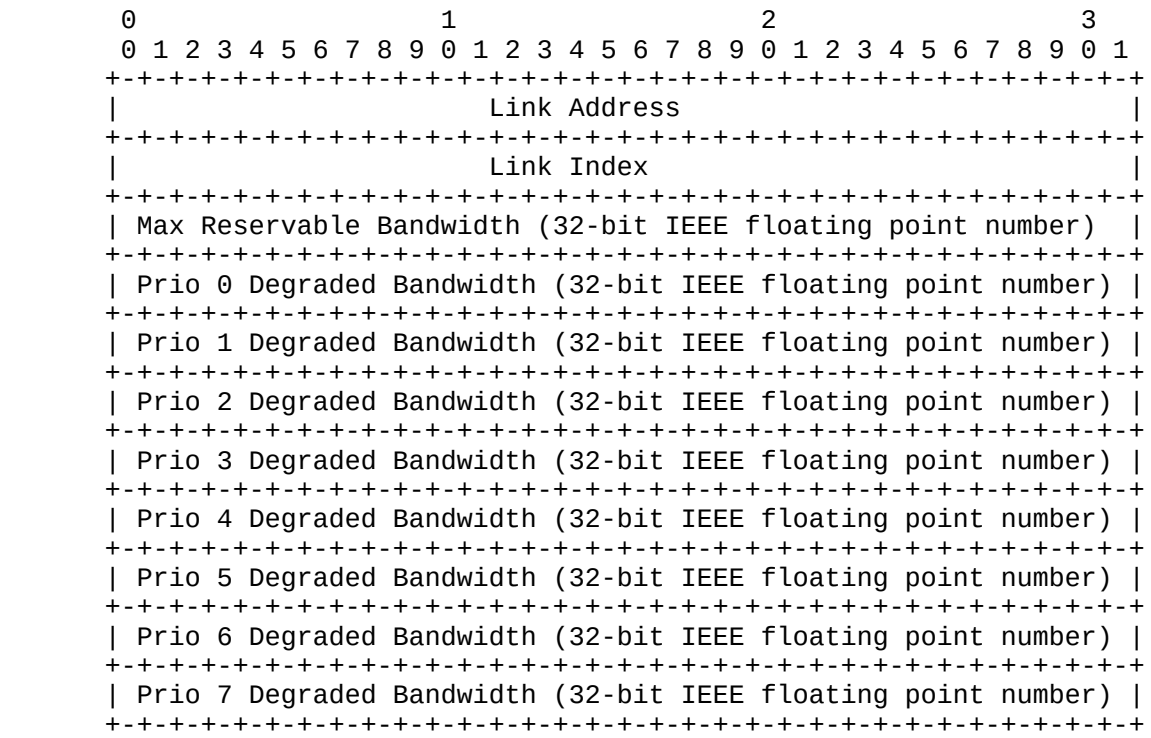
  - IPv6 address of the unavailable TE link.

* Link Index:

  - Index of the unavailable TE link.

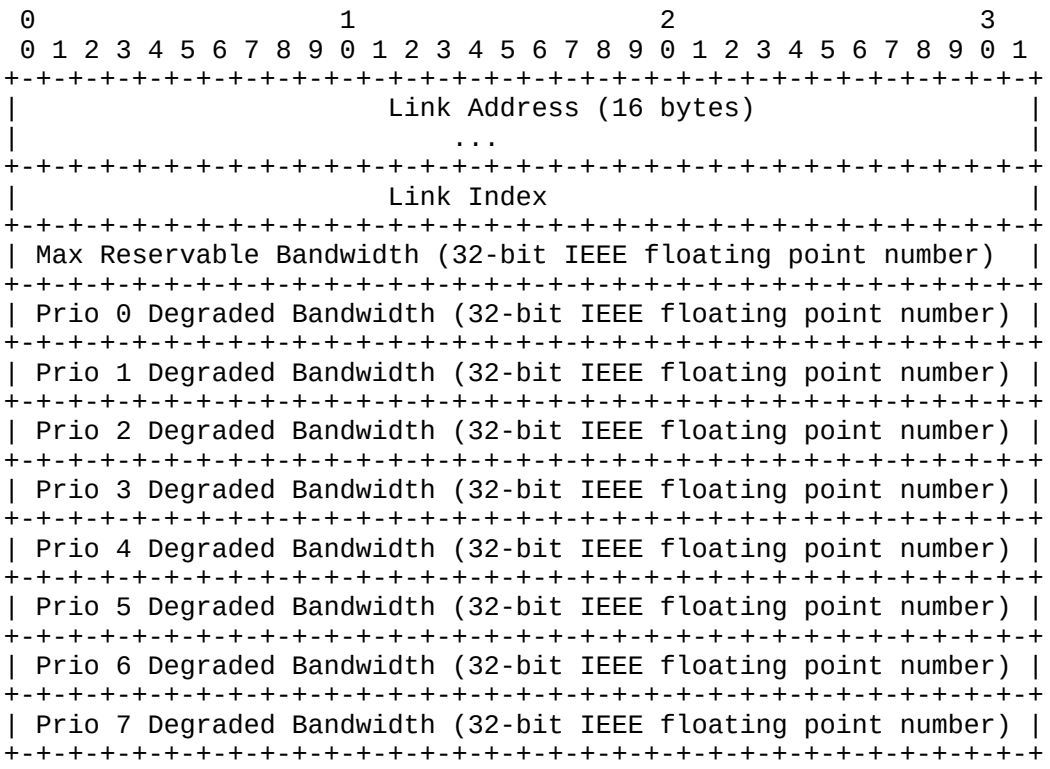
4.6.13. DEG_RESOURCE_SPEC

Class = DEG_RESOURCE_SPEC, DEG_RESOURCE_SPEC_Ipv4 C-Type = TBD



* Link Address:
  - IPv4 address of the degraded TE link.
* Link Index:
  - Index of the degraded TE link.
* Max Reservable Bandwidth:
  - Maximum reservable bandwidth on the degraded TE link
* Prio x Degraded Bandwidth:
  - Amount of bandwidth to be reduced for priority x.

Class = DEG_RESOURCE_SPEC, DEG_RESOURCE_SPEC_Ipv6 C-Type = TBD



* Link Address:
  - IPv6 address of the degraded TE link.
* Link Index:
  - Index of the degraded TE link.
* Max Reservable Bandwidth:
  - Maximum reservable bandwidth on the degraded TE link
* Prio x Degraded Bandwidth:
  - Amount of bandwidth to be reduced for priority x.

5. Protection

(To be added in a later revision.)

6. Graceful Restart

(To be added in a later revision.)

7. Security Considerations

(To be added in a later revision.)

8. IANA Considerations

(To be added in a later revision.)

9. Acknowledgments

The authors would like to thank Sudharsana Venkatraman for her input from discussions.

10. References

10.1. Normative References

[I-D.draft-kompella-teas-mpte]
Kompella, K., Jalil, L., Khaddam, M., and A. Smith, "Multipath Traffic Engineering", Work in Progress, Internet-Draft, draft-kompella-teas-mpte-01, 7 July 2025, <https://datatracker.ietf.org/doc/html/draft-kompella-teas-mpte-01>.

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

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, DOI 10.17487/RFC3209, December 2001, <https://www.rfc-editor.org/rfc/rfc3209>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

10.2. Informative References

[I-D.kompella-lsr-mptecap]
Kompella, K., "Multipath Traffic Engineering Capabilities", Work in Progress, Internet-Draft, draft-kompella-lsr-mptecap-00, 7 July 2025, <https://datatracker.ietf.org/doc/html/draft-kompella-lsr-mptecap-00>.

[RFC4875] Aggarwal, R., Ed., Papadimitriou, D., Ed., and S. Yasukawa, Ed., "Extensions to Resource Reservation Protocol - Traffic Engineering (RSVP-TE) for Point-to-Multipoint TE Label Switched Paths (LSPs)", RFC 4875, DOI 10.17487/RFC4875, May 2007, <https://www.rfc-editor.org/rfc/rfc4875>.

[RFC5440] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, DOI 10.17487/RFC5440, March 2009, <https://www.rfc-editor.org/rfc/rfc5440>.

[RFC8231] Crabbe, E., Minei, I., Medved, J., and R. Varga, "Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE", RFC 8231, DOI 10.17487/RFC8231, September 2017, <https://www.rfc-editor.org/rfc/rfc8231>.

[RFC8281] Crabbe, E., Minei, I., Sivabalan, S., and R. Varga, "Path Computation Element Communication Protocol (PCEP) Extensions for PCE-Initiated LSP Setup in a Stateful PCE Model", RFC 8281, DOI 10.17487/RFC8281, December 2017, <https://www.rfc-editor.org/rfc/rfc8281>.

Appendix A. Signaling Sequences - Examples

A.1. Initial Setup Sequence

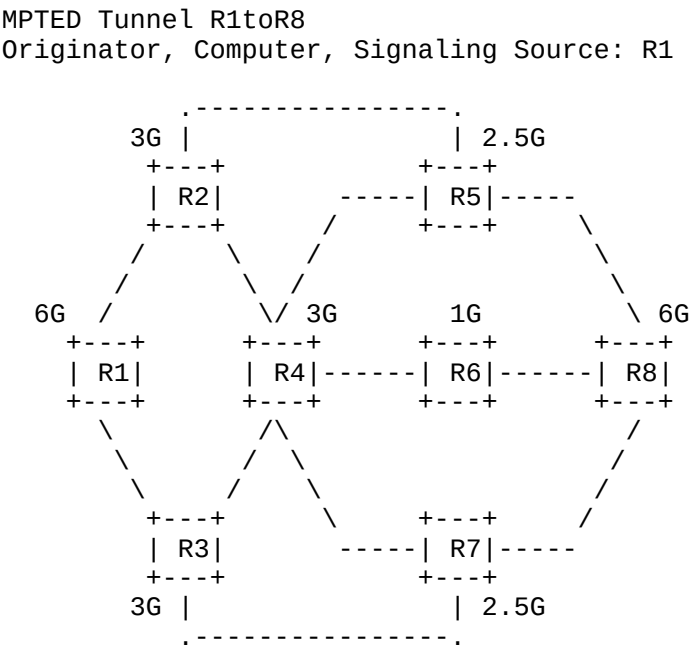


Figure 2: MPTED Tunnel Setup

* Initiation of setup sequence on MPTED tunnel signaling source, R1:

  - R1 sends an M-Path message to each junction node (R2, R3, R4, R5, R6, R7, and R8)

  - R1 processes the ingress JUNCTION, constructs a JSB, and waits for an M-Resv message to arrive from each JCT-NHOP (R2 and R3).

* M-Path message processing on transit junction nodes (R2, R3, R4, R5, R6, R7):

  - Each transit junction node processes the JUNCTION, constructs a JSB, and waits for an M-Resv message to arrive from each JCT-NHOP.

* M-Path message processing on egress junction node, R8.

  - R8 processes the JUNCTION and constructs a JSB.

- R8 sends an M-Resv message to each JCT-PHOP (R5, R6, and R7) with IMPLICIT NULL Label (3). -R8 sends an M-Notify message to R1, indicating that the junction processing is complete at R8.

* M-Resv message processing on transit junction nodes (R2, R3, R4, R5, R6, R7):

  - Each transit junction node waits until M-Resv messages are received from all available JCT-NHOPs and then:

    o Allocates a label for each JCT-PHOP and programs the corresponding labeled route.

    o Sends an M-Resv message to each JCT-PHOP with the corresponding allocated label.

    o Sends an M-Notify message to R1, indicating that the junction processing is complete on the node.

* M-Resv message processing on ingress junction node, R1:

  - R1 waist until M-Resv messages are received from all JCT-NHOPs (R2 and R3) and then:

  - Programs a route for the MPTED tunnel.

  - Notifies the signaling source (itself) that the junction processing is complete on the ingress node.

* M-Notify message processing on the signaling source:

  - The signaling source (R1) considers the setup sequence complete when confirmation of junction provisioning is received from all junctions.

    o If all the junctions indicate that the JUNCTION is "Up", then the status of the MPTED Tunnel is deemed "Up".

    o If all the junctions indicate that the JUNCTION is "Down", then the status of the MPTED Tunnel is deemed "Down".

    o In all other scenarios, the status of the MPTED Tunnel is deemed to be "Degraded".

A.2. Update Sequence - "In-Place"

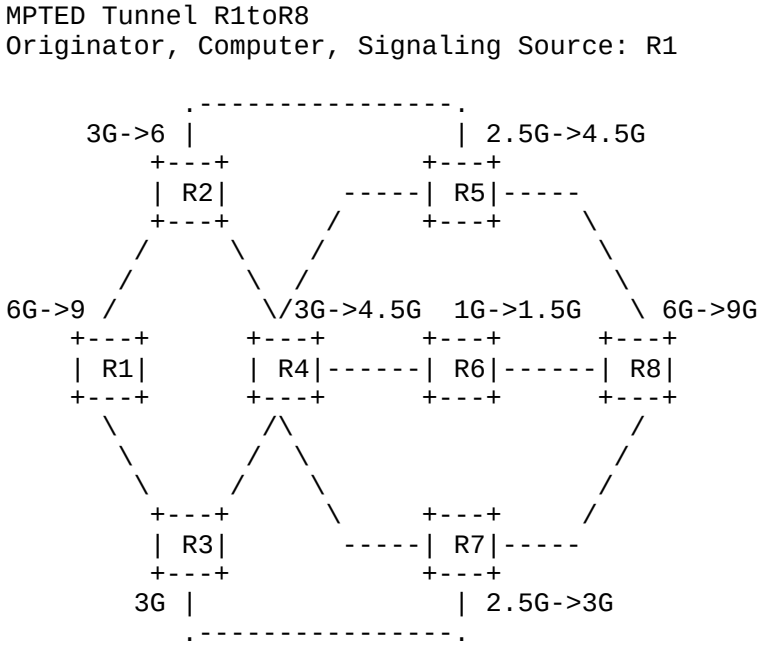


Figure 3: MPTED Tunnel Update - In-Place

* Initiation of update sequence on MPTED signaling source, R1:

  - R1 sends an M-Path message with a new version to each junction node (R2, R3, R4, R5, R6, R7, and R8)

  - R1 processes the updated ingress JUNCTION, updates the JSB, and waits for an M-Resv message to arrive from each JCT-NHOP (R2 and R3).

* M-Path message processing on transit junction nodes (R2, R3, R4, R5, R6, R7):

  - Each transit junction node processes the JUNCTION, updates the JSB, and waits for an M-Resv message with the new version to arrive from each JCT-NHOP.

* M-Path message processing on egress junction node, R8:

- R8 processes the JUNCTION and updates the JSB.

- R8 sends an M-Resv message with the new version to each JCT-PHOP (R5, R6, and R7)

- R8 sends an M-Notify message to R1, indicating that the junction update processing is complete at R8.

* M-Resv message processing on transit junction nodes (R2, R3, R4, R5, R6, R7):

  - Each transit junction node waits until M-Resv messages with the new version are received from all available JCT-NHOPs and then:

    o Reprograms the next-hops on the corresponding labeled route with updated load share (if needed).

    o Sends an M-Resv message with the new version to each JCT-PHOP.

    o Sends an M-Notify message to R1, indicating that the junction update processing is complete on the node.

* M-Resv message processing on ingress junction node, R1:

  - R1 waist until M-Resv messages with the new version are received from all JCT-NHOPs (R2 and R3) and then:

    o Reprograms the next-hops on the route for the MPTED tunnel with updated load share (if needed)

    o Notifies the signaling source (itself) that the junction update processing is complete on the ingress node.

* M-Notify message processing on the signaling source:

  - The signaling source (R1) considers the "in-place" update sequence complete when confirmation of junction update is received from all junctions.

A.3. Update Sequence - Add Junction

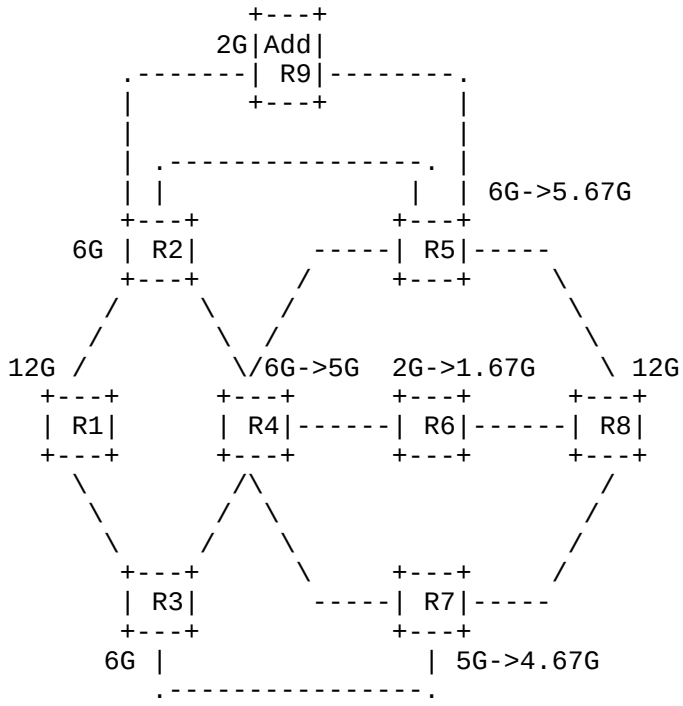


Figure 4: MPTED Tunnel Update - Add Junction

* Initiation of update sequence on MPTED signaling source, R1:

  - R1 sends an M-Path message with a new version to each junction node (R2, R3, R4, R5, R6, R7, R8, and the newly added R9)

  - R1 processes the updated ingress JUNCTION, updates the JSB, and waits for an M-Resv message to arrive from each JCT-NHOP (R2 and R3).

* Procedures on R3, R4, R6, R7, and R8 are the same as discussed in the example for "in-place update".

* Procedure on R9 is the same as discussed in the example for "initial setup".

* M-Path message processing on R2:

  - R2 processes the JUNCTION, updates the JSB, adds a new JCT-NHOP, and waits for an M-Resv message with the new version to be received from each JCT-NHOP (R4, R5, and R9).

* M-Path message processing on R5:

  - R5 processes the JUNCTION, updates the JSB, adds a new JCT-PHOP, and waits for an M-Resv message with the new version to be received from R8.

* M-Resv message processing on R5:

  - R5 allocates a label for JCT-PHOP 9 and programs the corresponding labeled route.

  - R5 sends an M-Resv message with the new version to each JCT-PHOP.

  - R5 sends an M-Notify message to R1, indicating that the junction update processing is complete on the node.

* M-Resv message processing on R2:

  - R2 waits until M-Resv messages with the new version are received from all available JCT-NHOPs and then:

    o Reprograms the labeled routes to include JCT-NHOP 9 in the list of next-hops.

    o Sends an M-Resv message with the new version to each JCT-PHOP.

    o Sends an M-Notify message to R1, indicating that the junction update processing is complete on R2.

* M-Notify message processing on the signaling source:

  - The signaling source (R1) considers the update sequence complete when confirmation of junction processing is received from all junctions.

Authors' Addresses

Kireeti Kompella
Juniper Networks
Sunnyvale, California 94089
United States of America
Email: kireeti.ietf@gmail.com

Vishnu Pavan Beeram
Juniper Networks
Sunnyvale, CA 94089
United States of America
Email: vbeeram@juniper.net

Chandra Ramachandran
Juniper Networks
Bengaluru
India
Email: csekar@juniper.net