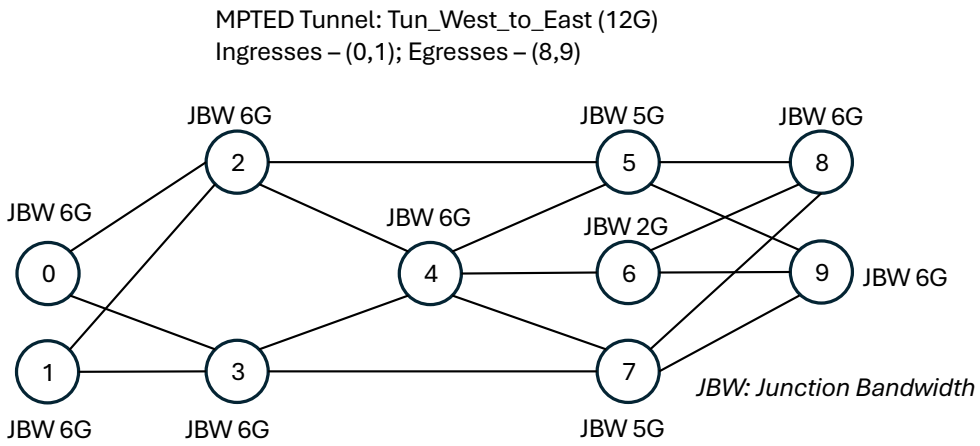
RSVP-TE Extensions for Multipath Traffic Engineered Directed Acyclic Graph Tunnels

draft-kbr-teas-mptersvp

Kireeti Kompella
Vishnu Pavan Beeram
Chandra Ramachandran

Introduction

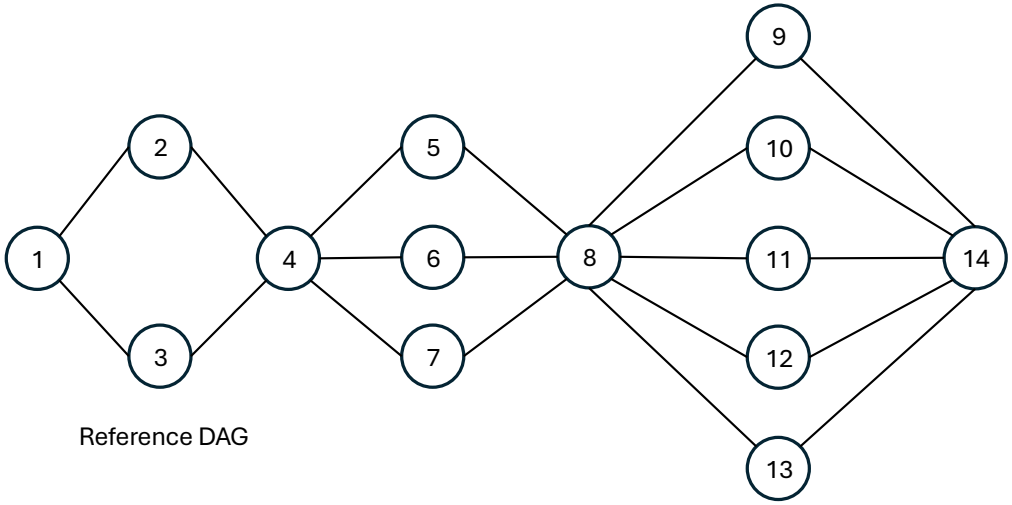
- An MPTED tunnel [I-D.draft-kompella-teas-mpte] is a Traffic Engineering (TE) construct that contains a constrained set of paths representing an optimized Directed Acyclic Graph (DAG) from one or more ingresses to one or more egresses.
  - The paths that make up an MPTED tunnel traverse a set of junction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



- [I-D.draft-kbr-teas-mptersvp] discusses the extensions to RSVP-TE for use as a signaling protocol to provision MPTED tunnels.
  - The current focus 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.

MPTED vs MPTEC Tunnels

- A Multipath Traffic Engineered Container (MPTEC) tunnel (widely deployed) contains multiple dynamically created and individually signaled single-path RSVP P2P tunnels.
  - Member tunnels are dynamically added and removed from the container tunnel at the ingress depending on the amount of traffic steered onto it.
  - The requirement to individually signal and maintain member LSP state can be a deterrent in specific scaled deployments.



Reference DAG

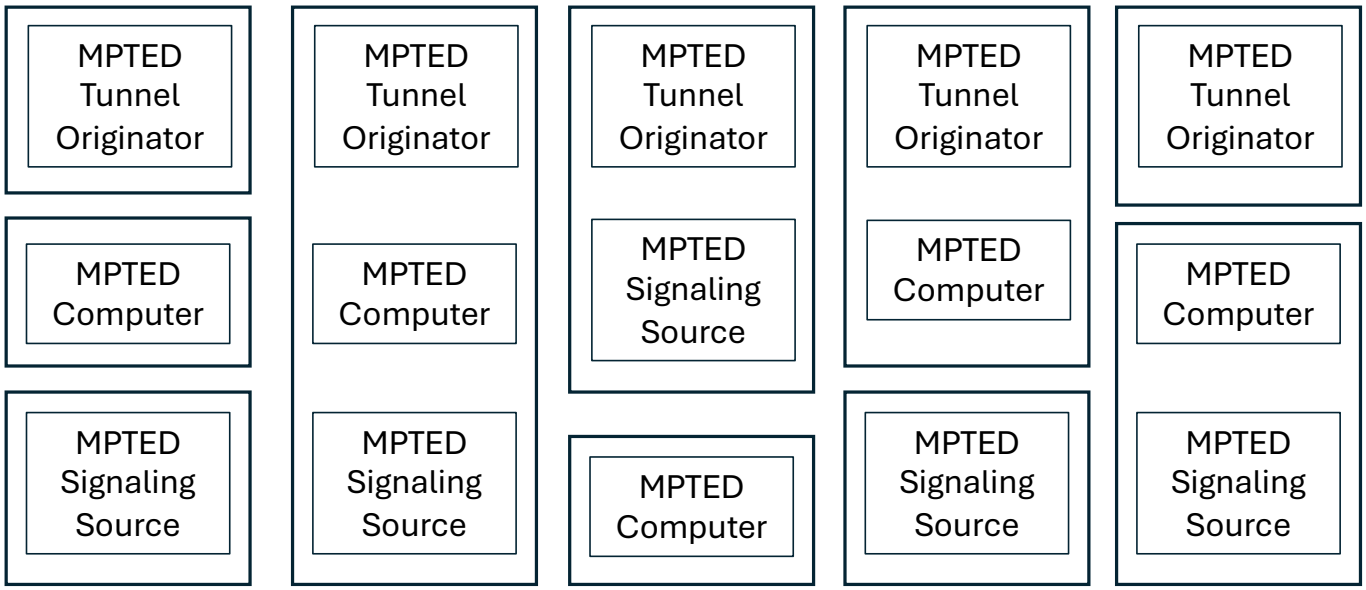
- Differentiators
  - With the MPTED tunnel, traffic is load-balanced across the next hops at each junction node in the DAG (in a weighted fashion), whereas with the MPTED tunnel, traffic is load-balanced only at the ingress node (and typically equally balanced among the next hops).
  - The amount of signaling needed to set up the tunnel is significantly less for the MPTED tunnel compared to the MPTEC tunnel.
  - An 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

Consider the reference DAG -

- *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.*
- *An MPTEC tunnel setup would result in 360 messages (180 Paths, 180 Resvs) on the wire, while an MPTED tunnel setup would result in 46 messages (13 M-Paths, 13- M-Notifys, 20 M-Resvs) on the wire*
- *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.*

MPTED Tunnels – Key Functions

- MPTED Tunnel Originator (TO)
  - Responsible for maintaining configuration and operational state for the tunnel
    - Identifier, Ingresses, Egresses, Constraints, Optimization objective
    - Identifier: <MPTED Originator ID (MPTED OID), MPTED ID>
- MPTED Signaling Source (SS)
  - Responsible for provisioning and maintaining junction state on each JUNCTION
    - Junction State Block (JSB) includes state for JSB-PHOPs and JSB-NHOPs
- These functions may be performed by one or more entities
- MPTED Computer (MC)
  - Responsible for computing an MPTE DAG that caters to the constraints and optimization objective
    - Computation result is a set of unordered elements called JUNCTIONs.
      - Each 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)



Key Concepts [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.
  - Not all changes to the DAG result in an immediate MPTED version change
    - In scenarios where only junction bandwidth or load-share on a junction nexthop is being updated on a sub-graph, the version associated with the impacted hops changes immediately
      - The MPTED version change is scheduled for later

- 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.
  - 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.
    - If only “additions” are being made, the update can be carried out in-place.
    - If the update involves any “deletion”, 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 in scenarios where a JUNCTION or a JCT-HOP is deleted.

Key Concepts [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.
- 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.

RSVP Signaling Messages for Junction Management

- (Signaling) Source to Junction (S2J) Messages
  - JunctionCreate
    - RSVP MPTED Path
  - JunctionUpdate
    - RSVP MPTED Path
  - JunctionDelete
    - RSVP MPTED PathTear (with or without CONDITIONS object)

- Junction to Source (J2S) Messages
  - JunctionNotify
    - RSVP MPTED Notify
  - ResourceNotify
    - RSVP Rsrc Notify

- Junction to Junction (J2J) Messages
  - Upstream (J2JU) Messages
    - JunctionNextHopReservation
      - RSVP MPTED Resv
    - JunctionDown
      - RSVP MPTED Notify

  - Downstream (J2JD) Messages
    - JunctionDelete – Conditional
      - RSVP MPTED PathTear (with CONDITIONS object)
    - JunctionNotReady
      - RSVP MPTED ResvErr

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>)

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.
  - Facilitates:
    - Ordered programming of labeled routes at each junction node on the DAG
    - Ordered admission control and bandwidth reservation on traversed TE links
    - Ordered addition/deletion of next hops when changing the shape of the DAG.
- 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)

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 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)

MPTED ResvErr (M-ResvErr) Message

- An M-ResvErr message is a J2JD message that is used by a junction node to notify its downstream junction node that it is not yet ready to accept the Resv message.
- The M-ResvErr message includes the following information:
  - MPTED tunnel identifier (SESSION Object)
  - MPTED tunnel instance identifier (VERSION Object)
  - Error Information
    - No matching junction state
  - Hop specific information
    - Hop identifier

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:
  - MPTED tunnel identifier (SESSION Object)
  - MPTED tunnel instance identifier (VERSION Object)
  - MTU (JUNCTION_STATUS Object)
  - Status (JUNCTION_STATUS Object)

  If the Status is "Degraded", the M-Notify message includes (additionally):
  - 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)

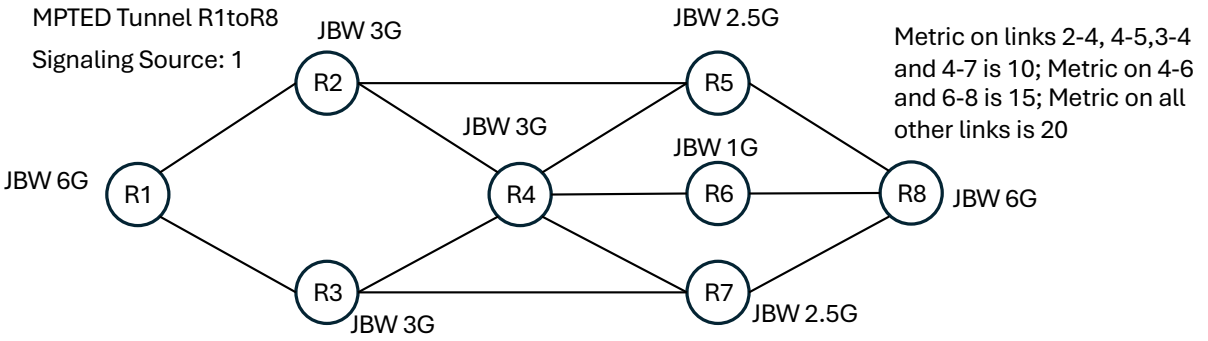
Resource Notify (RsrcNotify) Message

- 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.

Initial Setup Sequence

- 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:
    - Updates BW reservation on TE-links.
    - Allocates a label for each JCT-PHOP and programs the corresponding labeled route.
    - Sends an M-Resv message to each JCT-PHOP with the corresponding allocated label.
    - 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:
    - Updates BW reservation on TE-links.
    - 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.

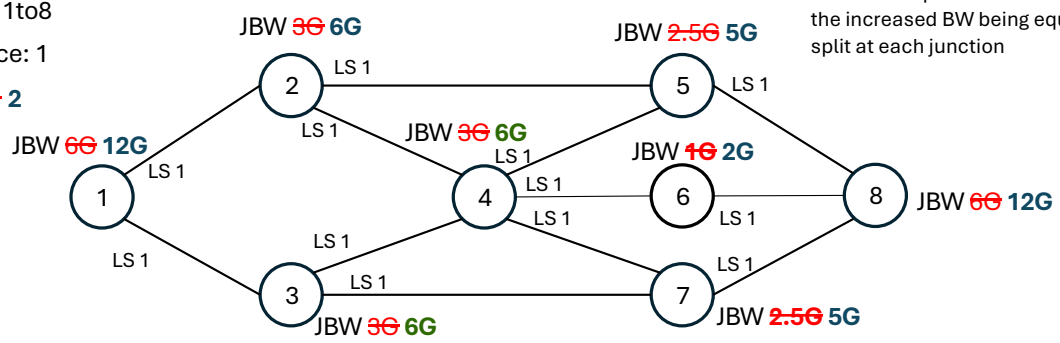
Update Sequence [1]: DAG (Full-Graph) In-Place Update – No change in DAG shape

- Initiation of update sequence on MPTED tunnel 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.

MPTED Tunnel 1to8

Signaling Source: 1

DAG Version: ~~1~~ 2



MPTED computation results in the increased BW being equally split at each junction

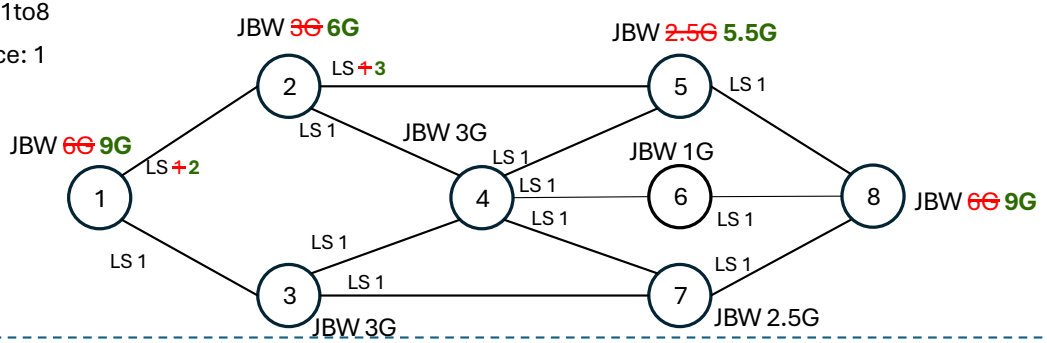
- M-Resv message processing on transit junction nodes (R2, R3, R4, R5, R6, R7):
  - Each transit junction node waits until M-Resv messages with the new version are received from all available JCT-NHOPs and then:
    - Updates BW reservation on TE-links.
    - Reprograms the next-hops on the corresponding labeled route with updated load share (if needed).
    - Sends an M-Resv message with the new version to each JCT- PHOP.
    - 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:
    - Reprograms the next-hops on the route for the MPTED tunnel with updated load share (if needed)
    - 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.

Update Sequence [2]: DAG [Sub-Graph] In-Place Update – No change in DAG shape

MPTED Tunnel 1to8

Signaling Source: 1

DAG Version: 1



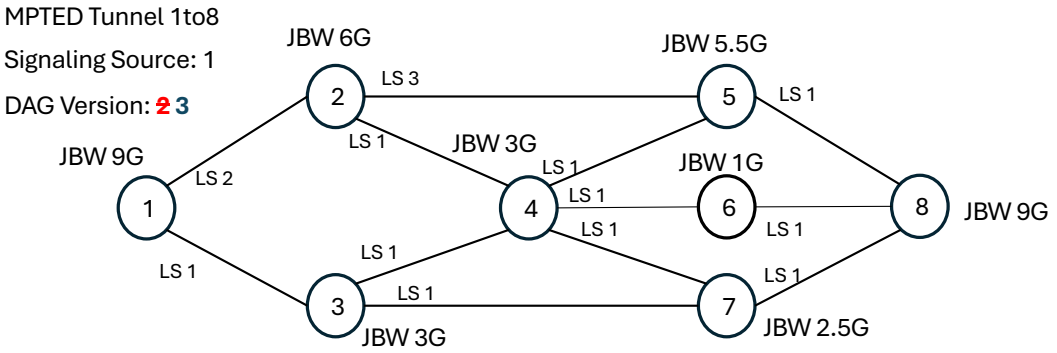
- MPTED tunnel BW needs to be updated to accommodate 9G (increased by 3G)
- MPTED computation results in the excess BW being placed on 1-2, 2-5 and 5-8 links
  - BW reservation needs to be updated only on these links
- Signaling involves changing versions for specific MPTED hops on 1, 2, 5 and 8; Rest of the hops in the MPTED remain unchanged

---

- Initiation of update sequence on MPTED tunnel signaling source (and ingress JCT), R1:
  - Sends an M-Path each to 2 (new version for JCT-PHOP 1 and JCT-NHOP 5), 5 (new version for JCT-PHOP 2 and JCT-NHOP 8), and 8 (new version for JCT-PHOP 5)
  - Updates JSB, moves JCT-NHOP 2 to "Update-In-Progress" state and waits for an M-Resv from 2
- M-Path message processing on transit JCT, R2:
  - Updates JSB, moves JCT-PHOP 1 and JCT-NHOP 5 to "Update-In-Progress" state and waits for an M-Resv from 5
- M-Path message processing on transit JCT, R5:
  - Updates JSB, moves JCT-PHOP 2 and JCT-NHOP 8 to "Update-In-Progress" state and waits for an M-Resv from 8
- M-Path message processing on egress JCT R8:
  - Updates JSB
  - Sends an M-Resv to 5 with the new "JCT-HOP version"
  - Sends an M-Notify to 1 confirming that the update is complete on junction 8
- M-Resv message processing on transit JCT, R5:
  - Updates BW reservation on TE-link 5-8
  - Moves JCT-NHOP 8 and JCT-PHOP 2 to "Up" state
  - Sends an M-Resv to 2 with the new "JCT-HOP version"
  - Sends an M-Notify to 1 confirming that the update is complete on junction 5
- M-Resv message processing on transit JCT, R2:
  - Updates BW reservation on TE-link 2-5
  - Updates labeled route with new load share
  - Moves JCT-NHOP 5 and JCT-PHOP 1 to "Up" state
  - Sends an M-Resv to 1 with the new "JCT-HOP version"
  - Sends an M-Notify to 1 confirming that the update is complete on junction 2
- M-Resv message processing on ingress JCT, R1:
  - Updates BW reservation on TE-link 1-2
  - Updates tunnel route with new load share
  - Moves JCT-NHOP 2 to "Up" state
  - Notifies signaling source
- M-Notify message processing on the signaling source:
  - The update sequence is deemed complete when the signaling source receives confirmation of junction provisioning from 1,2,5, and 8

Update Sequence [3]: DAG In-Place Update – Version Sync

- Initiation of update sequence on MPTED tunnel signaling source (and ingress JCT), R1:
  - 1 sends an M-Path each to 2, 3, 4, 5, 6, 7, and 8 (new DAG version).
  - Updates JSB, moves JCT-NHOPs to "Update-In-Progress" state and waits for an M-Resv from 2 and 3
- M-Path message processing on transit JCTs – R2, R3, R4, R5, R6, and R7:
  - Update JSB, move hops to "Update-In-Progress" state and wait for an M-Resv (with new DAG version) from each NHOP
- M-Path message processing on transit JCT, R8
  - Updates JSB
  - Sends an M-Resv each to 5, 6 and 7 with new DAG version
  - Send an M-Notify to 1 confirming that the update is complete on junction 8



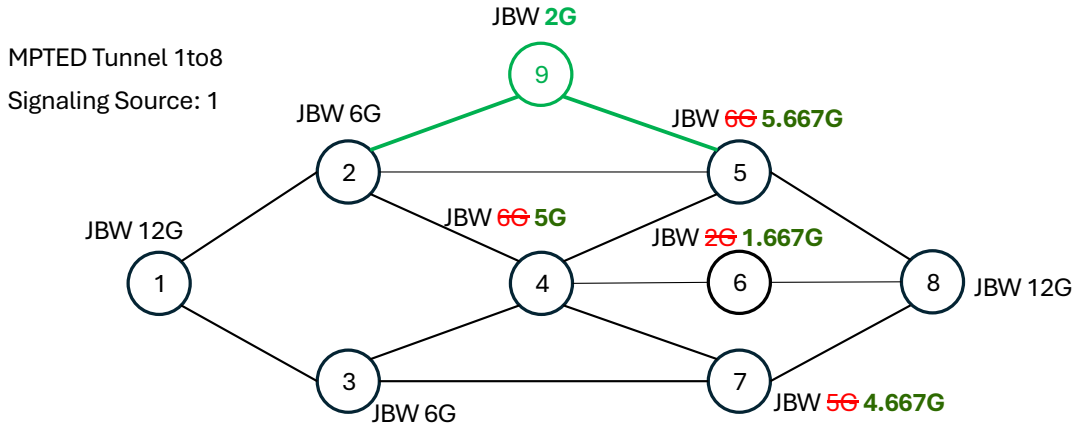
- A version sync operation for the MPTED tunnel is scheduled when a sub-graph update is performed and there is no version sync operation already scheduled.
- During a version sync operation, the DAG version is updated and made consistent across all junctions.

- M-Resv message processing on transit JCTs - R2, R3, R4, R5, R6, and R7
  - Each transit JCT waits until M-Resv messages with the new version are received from all available JCT-NHOPs and then:
    - Move JCT-Hops to "Up" state
    - Send an M-Resv to each PHOP
    - Send an M-Notify to 1 confirming that the junction update is complete

- M-Resv message processing on ingress JCT, R1
  - R1 waits until M-Resv messages with the new version are received from all JCT-NHOPs (R2 and R3) and then:
    - Moves JCT-Nhops to "Up" state
    - Notifies signaling source
- M-Notify message processing on the signaling source:
  - The update sequence is deemed complete when the signaling source receives confirmation of junction provisioning from all junctions.

Update Sequence [4] – DAG In-Place Update – JCT Addition

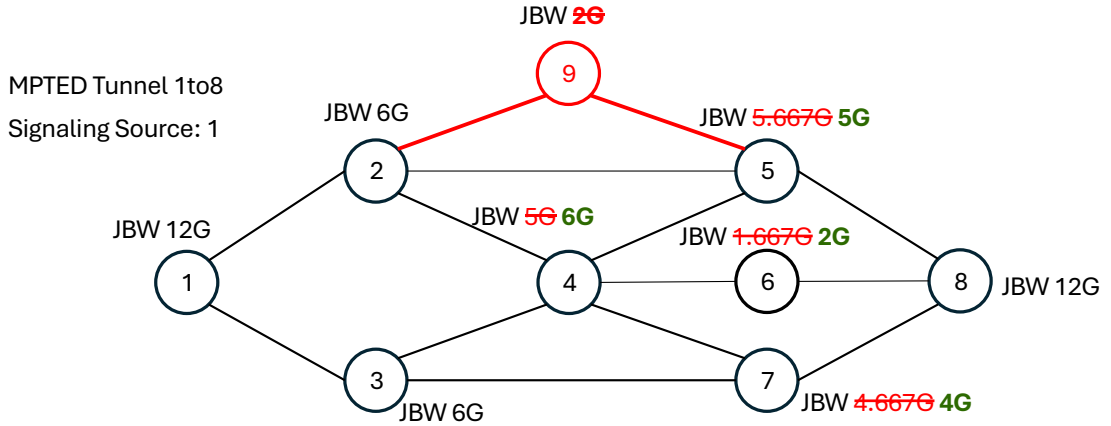
- Initiation of update sequence on MPTED tunnel signaling source (and ingress JCT), R1:
  - Sends an M-Path each to R2 (new JCT-NHOP 9, new versions for JCT-NHOPs 5 and 4), R4 (new versions for JCT-PHOP 5 and JCT-NHOPs 5, 6, and 7), R5 (new JCT-PHOP 9, new versions for JCT-NHOP 8 and JCT-PHOPs 2 and 4), R6 (new versions for JCT-PHOP 4 and JCT-NHOP 8), R7 (new versions for JCT-PHOP 4 and JCT-NHOP 8), R8 (new versions for JCT-PHOPs 5, 6, and 7), and R9 (new JCT).
- Procedures on R4, R6, R7 and R8 are the same as the procedures discussed previously for in-place update signaling sequence.
- Procedure on R9 is the same as discussed previously for initial setup sequence.
- M-Path message processing on R2
  - Update JSB, add a new JCT-NHOP for 9 and move it to ”Setup-In-Progress” state, move other JCT-NHOPs to “Update-In-Progress” state and wait for an M-Resv (with appropriate JCT-HOP version) from each NHOP



- M-Path message processing on R5
  - Update JSB, add a new JCT-PHOP for R9 and move it to “Setup-In-Progress” state, move other JCT-HOPs to “Update-In-Progress” state and wait for an M-Resv (with new JCT-HOP ver) from 8
- M-Resv message (from 8) processing on R5
  - Allocate a label for JCT-PHOP 9 and program corresponding labeled route
  - Send an M-Resv to each JCT-PHOP with new version
  - Send an M-Notify to R1
- M-Resv message processing on 2
  - Wait until M-Resvs from all NHOPs arrive and then -
    - Reprogram all labeled routes with next-hop 9 included in the list of next-hops
    - Send an M-Notify to 1
- M-Notify message processing on the signaling source:
  - The update sequence is deemed complete when the signaling source receives confirmation of junction provisioning from 2, 4, 5, 6, 7, 8, and 9

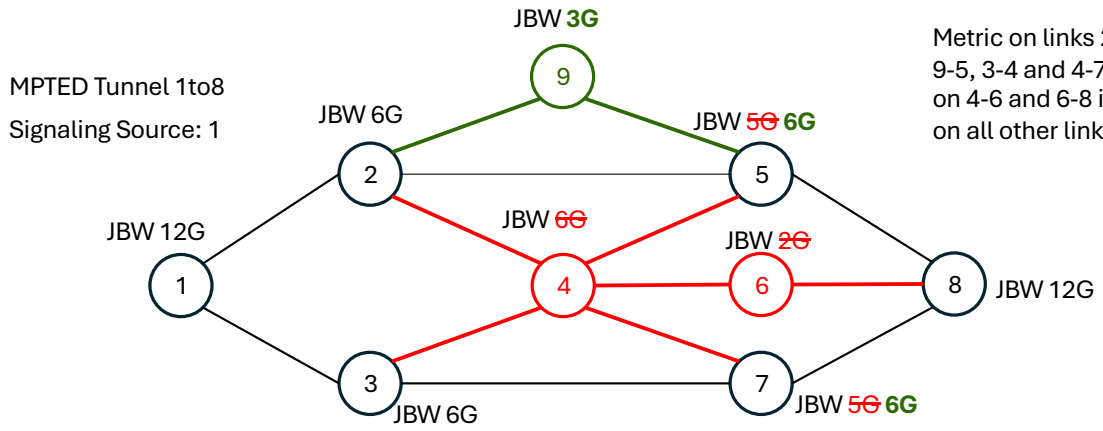
Update Sequence [5] – DAG MBB Update – JCT Deletion

- Initiation of update sequence on MPTED tunnel signaling source (and ingress JCT), R1:
  - Sends an M-Path (new DAG version) each to 2-8 (no message sent to 9)
  - Updates JSB and waits for an M-Resv each from 2 and 3
- M-Path/M-Resv message procedures on 1, 3, 4, 6, 7 and 8 are the same as discussed previously.
- M-Path message processing on R2
  - Update JSB, tag JCT-NHOP 9 as an element marked for deletion, and wait for an M-Resv (with new ver) from JCT-NHOPs 4 and 5.
- M-Path message processing on R5
  - Update JSB, tag JCT-PHOP 9 as an element marked for deletion, and wait for an M-Resv (with new ver) from 8
- M-Resv message (from 8) processing on 5
  - Send an M-Resv to JCT-PHOPs 2 and 4 with new version
  - Send an M-Notify to 1



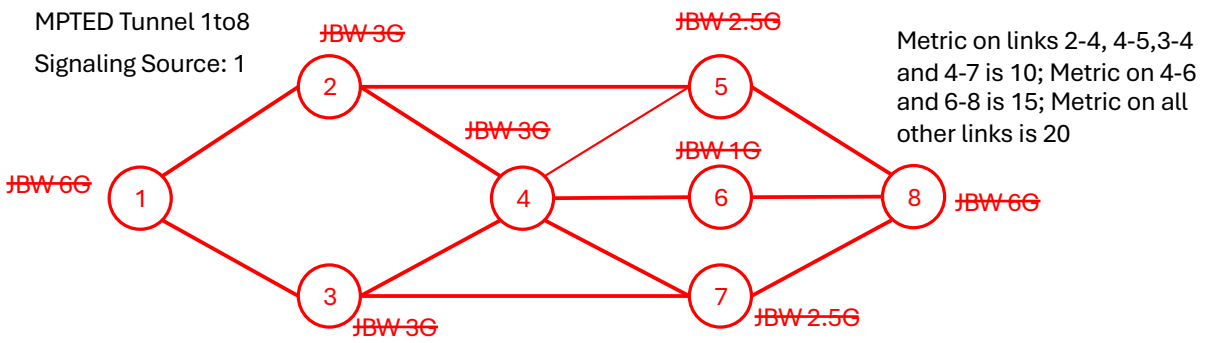
- M-Resv message processing on R2
  - Wait until M-Resvs from JCT-NHOPs 5 and 4 arrive and then –
    - Reprogram all labeled routes with next-hop 9 removed from the list of next-hops
    - Send an M-Resv to 1 with new version
    - Send an M-Notify to 1
- M-Notify message processing on signaling source (R1)
  - After receive confirmation of new DAG version provisioning from all JCTs, initiate a conditional M-PathTear from R1 for the old tunnel instance.
- Conditional M-PathTear message processing
  - Nodes that don't have JCT elements with matching version (1,3,4,6,7,8) simply forward the M-PathTear on each JCT-NHOP
  - Nodes that have JCT elements with matching version (2, 9, 5) delete the corresponding state and forward the M-PathTear downstream.
    - If no JCT-PHOP and JCT-NHOP is left, the JSB is deleted.
    - If there are multiple JCT-PHOPs with matching version, wait for M-PathTear to be received on all of them before deleting matching JCT-NHOPs.

Update Sequence [6]: DAG MBB Update - JCT Add/Delete



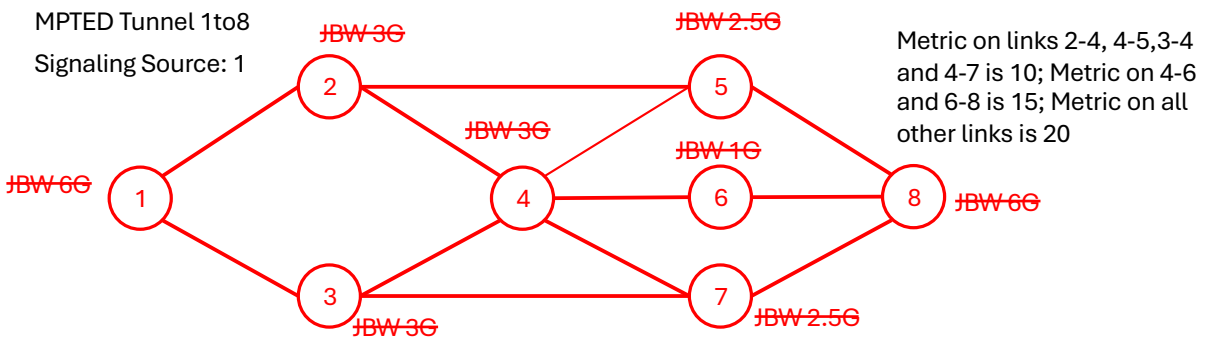
- Initiation of update sequence on MPTED tunnel signaling source (and ingress JCT), R1:
  - Sends an M-Path (new DAG version) each to 2, 3, 9, 5, 7, 8, and 9
  - Updates JSB and waits for an M-Resv each from 2 and 3
- M-Path message processing on R2:
  - Updates JSB, adds JCT-NHOP 9, tags JCT-NHOP 4 as an element marked for deletion, and waits for an M-Resv (with new ver) from 5 and 9
- M-Path message processing on R5
  - Updates JSB, adds JCT-PHOP 9, tags JCT-PHOP 4 as an element marked for deletion, and waits for an M-Resv (with new ver) from 8
- Message processing on R3, R7, R8, and R9 follows procedures discussed previously for add.
- M-Resv message processing on R5
  - Allocate a label for JCT-PHOP 9 and program corresponding labeled route
  - Send an M-Resv to JCT-PHOPs 2 and 5 with new DAG version
  - Send an M-Notify to R1
- M-Resv message processing on R2
  - Wait until M-Resvs from NHOPs 5 and 9 arrive and then –
    - Reprogram labeled route with next-hop 4 removed from the list of next-hops
    - Send an M-Resv to 1 with new version
    - Send an M-Notify to 1
- M-Notify message processing on signaling source (R1)
  - After receive confirmation of new DAG version provisioning from all JCTs, initiate a conditional M-PathTear from R1 for the old tunnel instance.
- Conditional M-PathTear message processing
  - Nodes that don't have JCT elements with matching version (1, 9) simply forward the M-PathTear on each available JCT-NHOP
  - Nodes that have JCT elements with matching version (2, 3, 4, 5, 6, 7, 8) delete the corresponding state and forward the M-PathTear downstream.
    - R4 deletes the JSB before forwarding the M-PathTear to its JCT-NHOPs

Delete Sequence [Option 1]



- Initiate Delete Sequence [1]
  - Delete local Junction State
  - 1 sends a S2J M-PathTear each to 2-8
- Recv S2J M-PathTear on 2-8
  - Delete Junction state
  - Send M-Notify to 1
- From the point of view of the tunnel signaling source, the tunnel is deleted as soon as the M-PathTear messages are sent out.
  - The receipt of the optional M-Notify messages (sent by the junction nodes in response to the M-PathTear messages) is logged by the tunnel signaling source and is used solely for debugging purposes.

Delete Sequence [Opt 2]



- Initiate Delete Sequence [2]
  - Delete local Junction State
  - Send M-PathTear (Conditional) to 2 and 3
- Recv M-PathTear on 2-8
  - Delete Junction state
  - Send M-PathTear (Conditional) to available JCT-NHOPs
  - Send M-Notify to 1
- From the point of view of the tunnel signaling source, the tunnel is deleted as soon as the M-PathTear messages are sent out at the ingress.
  - The receipt of the optional M-Notify messages (sent by the junction nodes in response to the M-PathTear messages) is logged by the tunnel signaling source and is used solely for debugging purposes.

Next Steps

- Protection
- Backwards Compatibility
- Graceful Restart

Thank You

kireeti.ietf@gmail.com
vbeeram@juniper.net
csekar@juniper.net