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

K. Kompella
Juniper Networks
L. Jalil
Verizon
M. Khaddam
Cox Communications
A. Smith
Oracle Cloud Infrastructure

7 July 2025

Multipath Traffic Engineering
draft-kompella-teas-mpte-01

Abstract

Shortest path routing offers an easy-to-understand, easy-to-implement method of establishing loop-free connectivity in a network, but offers few other features. Equal-cost multipath (ECMP), a simple extension, uses multiple equal-cost paths between any two points in a network: at any node in a path (really, Directed Acyclic Graph), traffic can be (typically equally) load-balanced among the next hops. ECMP is easy to add on to shortest path routing, and offers a few more features, such as resiliency and load distribution, but the feature set is still quite limited.

Traffic Engineering (TE), on the other hand, offers a very rich toolkit for managing traffic flows and the paths they take in a network. A TE network can have link attributes such as bandwidth, colors, risk groups and alternate metrics. A TE path can use these attributes to include or avoid certain links, increase path diversity, manage bandwidth reservations, improve service experience, and offer protection paths. However, TE typically doesn’t offer multipathing as the tunnels used to implement TE usually take a single path.

This memo proposes multipath traffic-engineering (MPTE), combining the best of ECMP and TE. The multipathing proposed here need not be strictly equal-cost, nor the load balancing equally weighted to each next hop. Moreover, the desired destination may be reachable via multiple egresses. The proposal includes a protocol for signaling MPTE paths using various types of tunnels, some of which are better suited to multipathing.

Status of This Memo

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

This Internet-Draft will expire on 8 January 2026.

Copyright Notice

Table of Contents

1. Introduction

Operators managing traffic within their networks have several tools, among them:

1. Equal-cost Multipath (ECMP): balance traffic along multiple paths. This yields some resilience and some traffic management, as traffic can be load-balanced across multiple paths. To use ECMP effectively, one may have to adjust link metrics to allow multiple paths to have the same overall distance.

2. Traffic Engineering (TE): state constraints for a path from an ingress router to an egress router, and let a path computation engine compute it. This gives much greater control over the nodes and links traversed, but is usually limited to finding a single path from ingress to egress [RFC2702].

3. Multi-egress: allow traffic from an ingress router to a destination dst to use several egress routers, all of which have routes to that destination. dst may be an Internet prefix [RFC4271], a VPN prefix [RFC4364], an EVPN address [RFC7432], a VPLS site [RFC4761], [RFC4762] or some other service destination. For BGP-signaled destinations, this requires that the BGP tie-breaking algorithm yield multiple results (rather than a single one), all of which become candidates for egress.

4. Multi-ingress: consider multiple ingress routers as "equivalent" with respect to some of the traffic they sent to one or more egress routers. For example, an eBGP peer router or a VPN site may be multi-homed to several ingress routers, all of which would send such traffic to the same set of egress routers.

[RFC2702] describes requirements for MPLS-based TE, and thus is relevant to this memo. At the same time, the authors appear to believe that one can either have TE or multipathing, but not both. This is further emphasized by the notion of a Label Switched Path, which is used to implement MPLS-based TE. RSVP-TE ([RFC3209]), the protocol designed to meet the requirements of [RFC2702], builds a single path from one ingress to one egress (for unicast traffic).

In order to satisfy the constraints, TE often uses non-shortest paths. To do so without looplng packets, a tunnel is used. Such tunnels have to be signaled. RSVP-TE is a signaling protocol for MPLS-based tunnels.

In this memo, we introduce a new tool: multipath TE (MPTE). This allows an operator to specify constraints for paths (as in TE), specify multiple ingresses and egresses, and use multiple paths from ingress to egress. Effectively, MPTE combines the advantages of the four tools above. The resulting set of paths from ingresses to egresses is a Directed Acyclic Graph (DAG), here called an MPTE DAG or MPTED. Finally, this memo allows the use of multiple types of tunnels. The main contribution of this memo is the notion of a (multipath) unicast tunnel across an MPTED, called an MPTE tunnel or MPTET, and an overview of how they are created. Protocols for provisioning such tunnels will be specified in companion documents. Another companion document defines how to distribute MPTE capabilities in an IGP so that entities computing MPTEDs can know which nodes to include in the DAG.

1.1. Terminology

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.

1.1.1. Definition of Commonly Used Terms

This section provides definitions for terms and abbreviations that have a specific meaning to the MPTE protocol and that are used throughout this memo.

constraints: desired properties of paths between ingresses and egresses.

constrained shortest path first (CSPF): A modification to SPF to take into account TE constraints.

directed acyclic graph (DAG): a directed graph that has no cycles. The result of a multipath SPF or CSPF computation is a DAG.

directed graph: a set of nodes and directed links. A network is represented by a directed graph.

egress: an end node of an MPTE DAG.

equal-cost multipath (ECMP): a DAG consisting of shortest paths from an ingress to an egress.

flow-aware load balancing (FALB): load balancing that maps packets that belong to a given flow onto the same path. What consitutes a flow depends on the type of traffic; for IP traffic, a flow is typically defined by the 5-tuple <source IP, dest IP, protocol type, source port, dest port>.

ingress: a starting node of an MPTE DAG.

label-switched path (LSP): an MPLS tunnel from an ingress to one or more egresses.

link: A (directed) edge between two nodes. A pair of nodes may have 0 or more links between them. A link between nodes u and v will be denoted by (u, v, i), where i is u’s oif for the link. A link may have associated attributes, in particular, a metric.

load balancing (LB): a method whereby traffic to an egress is distributed among multiple next hops at each node along the DAG.

metric: a positive number describing the contribution of a link to the oveall path length.

MC: MPTED computer: the entity computing the MPTED, typically the ingress (if there is a single ingress) or a Path Computation Element

MPTE: multipath TE with path constraints (including a slack) using nECMP paths from an ingress to one or more egresses.

MPTED: an MPTE DAG resulting from CSPF-type computation on MPTE constraints.

MPTEP: MPTE protocol: the protocol used to signal MPTEDs.

MPTET: MPTE tunnel: the forwarding entity associated with an MPTED over which traffic is sent.

non-equal-cost multipath (nECMP) (generally qualified by "with slack s"): a DAG of paths from an ingress u to an egress v that are within s of min(u, v).

node: a vertex of a graph. A node may have associated attributes.

outgoing interface (oif): a unique number (oif) assigned by a node for each outgoing link it has.

Path Computation Element (PCE): an entity that capable of performing CSPF on behalf of another node, the path computation client.

path length: the sum of the metrics of the links that constitute path p, denoted by len(p)

shared risk group (SRG): nodes and/or links that share "risk" (e.g., have a common power feed, or use a common fiber conduit)

shortest path: a path between a pair of nodes u, v with minimum length. The set of shortest paths between u and v is a DAG, denoted by sp(u, v). The length of a shortest path from u to v is denoted by min(u, v)

shortest path first (SPF): an algorithm for computing the shortest path DAG from an ingress to an egress; typically refers to Dijkstra’s algorithm for computing shortest paths between a given pair of nodes, or pairwise between all nodes.

signaling source (SS): an entity responsible for signaling an MPTET

slack: a path p from u to v has slack s if len(p) = min(u, v) + s.

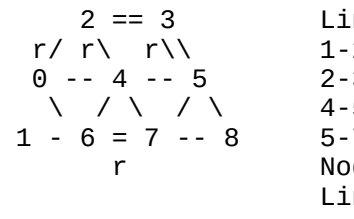
traffic engineering (TE): a methodology for mapping traffic trunks to given paths or DAGs across a network.

traffic trunk: a unidirectional aggregate of traffic flows from an ingress to a set of egresses that is treated identically in the forwarding plane.

tunnel originator (TO): entity having the specifications of the MPTET

2. Overview

Consider Figure 1:



Link Metrics (symm): 0-2: 100; 0-4: 200; 0-6: 110
1-2 (not shown): 110; 1-4 (not sh): 100; 1-6: 100
2-3: (100, 100); 2-4: 100; 3-5: (100, 110)
4-5: 100; 4-6: 110; 4-7: 50
5-7: 100; 5-8: 10; 6-7: (100, 110); 7-8: 50
Node pairs 2-3, 4-5 and 6-7 each have two links.
Links marked with 'r' have color red.

Figure 1: Network 1

2.1. Multipathing

2.1.1. ECMP (slack 0) from node 0 to node 5

There are 4 ECMP paths from node 0 to node 5:

1. 0-2=3-5 (2 paths)

2. 0-2-4-5

3. 0-4-5

These 4 distinct paths all have length 300.

2.1.2. nECMP from node 0 to node 5 with slack 10

There are 7 nECMP paths with slack 10 to node 5:

1. 0-2=3=5 (4 paths)

2. 0-2-4-5

3. 0-4-5

4. 0-6-7-5

These 7 paths have lengths 300 or 310. Thus, allowing nECMP paths a slack of 10 has yielded 3 additional paths, which provide increased diversity and load balancing, and possibly decreased congestion.

2.1.3. Multipathing from node 0 to egresses {5, 8}

If, for some traffic trunk that starts at node 0, nodes 5 and 8 are equally good as egresses, then one can compute an ECMP DAG from 0 to {5, 8}; this yields 4 paths to 5 and 6 paths to 8, for a total of 10 paths this traffic trunk can take. Similarly, a nECMP DAG to {5, 8} with slack 10 has 15 paths, whereas one with slack 5 has the same 10 paths as with slack 0.

2.1.4. MPTED from ingresses {0, 1} to egresses {5, 8}

If traffic from node 0 to nodes {5, 8} and from node 1 to nodes {5, 8} have common characteristics, it may make sense to compute a single DAG from {0, 1} to {5, 8}. Doing so allows the operator to view this entire DAG as one logical entity; a nice side benifit is reduced control and data plane state due to state sharing.

2.2. Load balancing

Nodes in a netword have a Forwarding Information Base (FIB). A FIB maps a packet's destination address da to one or more "next hops". When a packet with address da arrives at n, n sends the packet to one of the next hops. n typically will distribute packets in a given ratio among the next hops. This is load balancing.

The main goal of ECMP/nECMP is to supply as many nodes as possible in the MPTED with multiple next hops on which to forward the traffic trunk. At such nodes, traffic belonging to the trunk can be distributed among the next hops instead of going to a single next hop. This has the potential to reduce congestion and provide better utilization of available links.

2.2.1. Flow-aware load balancing

When load balancing packets from a traffic trunk, it is often required that packets from a given flow be sent to the same next hop. This improves the probability of in-order delivery of packets in that flow, which is important for certain types of traffic. This is called flow-aware load balancing (FALB). The most common flow in IP traffic is defined by a 5-tuple consisting of the source IP address, the destination IP address, the protocol, the source port and the destination port. A 16- or 20-bit hash of this 5-tuple is called the packet's entropy.

There are two common ways to achieve FALB of IP traffic. One is to do a "deepish" packet inspection (dPI), find the relevant 5-tuple, and use that to compute the packet's entropy. The entropy is then used to ensure that packets in the flow are sent to the same next hop. This memo suggests sending TE traffic over a tunnel (see {tunnels}); this makes the identification of IP flows expensive and error-prone.

Another way of accomplishing this is to insert the entropy in the tunnel header. Many of the tunnels suggested in this memo have such a field. The ingress is in a good position to identify flows, and, when encapsulating the packet into the tunnel, can insert the entropy in the header. The heavy lifting of identifying flows is thus placed

on the ingress. Transit nodes can simply use the entropy field to correctly map packets in a flow to the same next hop, thus ensuring FALB.

2.2.2. Per-packet load balancing

FALB is often required and is a good default behavior, especially as end applications may be expecting packets in a flow to be delivered in order. However, FALB has the issue that it attempts (statistically) to place roughly the number of flows in the given ratio on the outgoing links; that may not place traffic in the same ratio, as flows need not carry the same traffic. In some cases (typically when configured to), one can do per-packet load balancing (PPLB), meaning that load balancing is no longer flow aware. This can be done when the end applications do not require packets in a flow to be in order, or if some (bookended) devices outside the network put the packets back in order before delivering them to the applications (typically by addind a sequence number). When feasible, PPLB gives much better load distribution, and is currently the subject of investigation, implementation and standardization.

One can achieve this by configuring each router in the DAG to do PPLB for the traffic trunks in the DAG, or more simply by the ingress router assigning entropy at random to the traffic it places in the DAG. The latter approach keeps the decision of which DAGs (and corresponding traffic trunks) should be flow-aware and which not at the ingress; all other nodes simply do what the entropy fields tells them to do.

2.3. Constraints

Constraints are an intent-based specification of acceptable paths that a traffic trunk may take from ingress to egress(es). Constraints are thus an abstract way to control the resources that a particular traffic trunk uses.

One way to do this is to add "resource class attributes" or "colors" [RFC2702] to links, and then specify "include" and "exclude" sets. An include set means that all links that a path traverses must contain at least one element of the include set. An exclude set means that no link in the path can contain any color from the exclude set.

Another way is to specify a (maximum) bandwidth that a traffic trunk can carry. This means that all links in the path must have that much available capacity. Packets exceending the bandwidth can forwarded normally, marked as droppable, or dropped.

Let’s add some simple constraints to our DAG. We associate the color red to one of the links from B to C, and to the shorter of the links from F to G. Then, we constrain the paths to "exclude red", meaning avoid links with color red. This yields the following:

* ECMP from node 0 to node 5 with constraints "include red or blue" yields a single path.

2.4. Protection

One very useful aspect of TE is the ability to specify that a path must be link- or node- or shared-risk-disjoint from another path. That means that the two paths do not have links or nodes or "shared risk groups". Additionally, one can build protection paths for an existing path to protect against link or node failures [RFC4090]. This is especially important as TE currently takes a single path through the network, meaning that a link or node failure will result in dropped traffic until the TE path is restored.

While not quite as crucial in the case of an MPTED, since ideally, there will be multiple nexthops at each node, there will be cases where a node has a single next hop, or all next hops share a common failure mode. Identifying these cases and building protection paths for such nodes will be described in a future version of this memo.

2.5. Tunnels

The shortest path first algorithm [SPF] is an easy-to-implement and very efficient algorithm whereby all routers in a network can agree on the path that a packet to a particular destination should take. That means, if all routers are agreed (roughly) on the topology and metrics of the network, they will forward packets in a loop-free manner to all destinations -- without the need for signaling or tunnels. However, an MPTED will not contain the same paths -- some paths may be rejected as they don’t conform to the constraints; other paths may be used even though they are not shortest paths. Thus, to route packets in a traffic trunk over a computed MPTED, a tunnel is typically used. This tunnel will have to be signaled to the MPTED nodes. The tunnel may be MPLS- or IP-based.

A few things are important about tunnels: whether they carry an entropy field (EF), whether they have a "discriminator" (D) that allows multiple tunnels between an ingress-egress pair, whether they allow multiple egresses (ME), and whether they allow multiple ingresses (MI). These will be discussed in the description of the tunnels below.

In the memo, we consider the following tunnel types:

1. IP-in-IP: [RFC2003] encapsulation allows the creation of an "outer" IP header to carry a payload packet (which is typically an IP payload). The outer IP header's protocol field indicates the "protocol" of the inner payload packet. The outer header of IP-in-IP tunnel doesn't contain an EF; transit nodes can either spray packets across outgoing next hops, attempt to do dPI, or use the same next hop for all packets. To accommodate ME, the egresses have to have the same (anycast) IP address which would be used as the destination IP of the tunnel. MI is not possible.

2. GRE: Generic Routing Encapsulation. We include in this definition [RFC2784] and [RFC2890] with the Key Present (bit 2) set to 0. This is similar to IP-in-IP; however, the payload is not required to be IP. There is no EF in the header. D, ME and MI same as for IP-in-IP.

3. GRE-E: GRE with Key Present; the Key value is the EF. D, ME and MI same as for IP-in-IP.

4. GRE6: GRE with IPv6 addresses. The entropy is carried in the Flow Label field of the IPv6 header. D, ME and MI same as for IP-in-IP.

5. G-in-U: GRE-in-UDP [RFC8086]. The UDP source port is the EF; the GRE Key, if present, can be ignored from a load balancing point of view. D, ME and MI as in IP-in-IP.

6. MPLS-in-UDP [RFC7510]. The UDP source port is the EF; D, ME and MI as in IP-in-IP.

7. SigLab (signaled label switching). The labels to be used are signaled. Signaling proceeds from egress(es) to ingress(es). An entropy label can be used as the EF. At each node, a different label is used for each MPTED; this is the discriminator. ME and MI are both allowed.

8. StatLab (static label). A single statically-assigned label defines the tunnel throughout the MPTED. Here, a block of MPLS labels is given to a label allocator; these labels MUST NOT be allocated by any node in the network. EF, D, ME and MI are as for SigLab. The MPTED computer (MC) must interact with the allocator when creating or deleting an MPTED.

2.6. Backward Compatibility

Introducing a new idea to the network (and thus new protocols, new extenstion and new software) is typically done incrementally. Thus, in a network transitioning to MPTE, there will be some nodes that are MPTE-capable, and others that are not.

In Figure 1 above, if node 4 is not MPTE-capable, it can either be left out of the MPTED, or a "classical" tunnel can be constructed from (say) node 2 to node 5, allowing hybrid paths 0-2-(4)-5 and 0-2-(4)-5-8 for a DAG from {0} to {5, 8}. The signaling specs will say whether this is possible, and if so, how it can be achieved.

3. Operation

The starting point in building an MPTE DAG is to define the properties of a traffic trunk from ingress to egress. Examples include "BGP destinations with community xyz" or "gold class traffic belonging to VPN foo". Next, define a set of constraints that capture the types of paths permissible for this traffic trunk. These include a metric to minimize (perhaps with slack); this could capture delay or fiber length, link colors, shared risk groups (SRGs) and bandwidth. The desired outcome is an MPTED into which the traffic trunk can be mapped.

An MPTED is specified by defining:

1. a (non-empty) set of ingresses

2. a (non-empty) set of egresses

3. the metric to use and the slack

4. path constraints

5. whether or not the MPTED is "strict".

An MPTED is strict if all paths from all ingresses to all egresses are within slack of the shortest path. An MPTED is loose if all paths from a given ingress I to a given egress E are within slack of each other, but paths from I to a different egress F may not be within slack of the paths to I.

Computation (possibly using a variant of CSPF) of an MPTED is done by the MC, which is either an ingress or a PCE [RFC4655]. (This memo does not specify such an algorithm.) Signaling primarily occurs between the MC and each junction node. Auxiliary signaling may occur between a junction node and its phops.

3.1. MPTED

In this memo, a node is identified by its (16-octet) IPv6 loopback address. A link from node u to node v is identified by u's loopback address and its (4-octet) outgoing interface index (oif), a unique identifier for the link allocated by u. oifs are usually exchanged in the TE extensions of an IGP. (A link also has a (4-octet) incoming interface index, the iif. For neighbors u and v, the correlation between u's oif and v's iif is typically done by the IGP. iifs are not used in this memo.) For now, this memo only deals with point-to-point links; a future revision will describe the use of multi-access links.

An MPTED is identified by a unique (4-octet) ID (the MID) assigned to the MPTED by the MC. As an MPTED can change over its lifetime, it is assigned a version number starting at 0 and incremented every time the MPTED is recomputed. Thus, a full MPTED ID (the FID) consists of <MC, MID, version>.

An MPTED consists two or more "junction nodes". A junction node can have one of five types:

1. a pure ingress node has zero incoming links and one or more outgoing links in the MPTED. Traffic routed on a MPTED enters at the ingress.

2. a pure egress node has one or more incoming links and zero outgoing links in the MPTED. Traffic routed on a MPTED leaves at an egress.

3. a transit ingress node where traffic can either enter the MPTED or arrive from another ingress node to continue on in the MPTED.

4. a transit egress node where traffic can either exit the MPTED or go on to another egress node.

5. a "regular" junction node has one or more incoming links and one or more outgoing links. Traffic does not enter or leave at such a node: it comes from a phop and goes to an nhop.

A junction node v consists of v, its previous hops (phops) and its next hops (nhops). A phop is specified by an incoming link of v: (u, v, oif1); an nhop by an outgoing link of v: (v, w, oif2). Note that, since links are point-to-point, it is sufficient to specify (u, oif1) ((v, oif2)) for a phop (nhop, respectively). The nodes u (and w) are loosely referred to as a phop (and nhop) of v, although strictly speaking the link should be included. A pure ingress has no phops and a pure egress has no nhops.

The MPTED is broken down into a set of junction nodes. A junction node v is specified by:

1. bandwidth (coming in to and going out of v)

2. a list of phops of v

3. a list of nhops of v, with corresponding load balancing shares

3.2. Tunnel Provisioning

A designated entity, the Tunnel Originator (TO), is given the specifications of the MPTET: the ingresses, the egresses and the constraints. The TO is typically one of the tunnel ingresses or a PCE. The TO sends the tunnel specification to the MC. The MC computes the MPTED (as a list of junctions) and returns this to the TO. The TO then sends the list of junctions to the Signaling Source (SS) which provisions the tunnel.

Note that TO, MC and SS are functional blocks; they may reside on separate nodes or co-reside on the same node. For example, a single node may be the TO and SS but decide to delegate computation to a (remote) PCE. This node then gets the results via PCEP and signals the tunnel. Other permutations are possible.

3.3. Signaling Overview

The SS signals the creation or update of an MPTE tunnnel by sending to each junction node v a JUNCTION message consisting of:

1. the MPTET ID

2. the junction node specification

3. the tunnel type

4. some flags

After v parses this specification, for all tunnel types other than SigLab, it installs FIB state for the junction.

For tunnel type SigLab, v allocates an incoming MPLS label L_u for each phop u, and sends a LABEL message to u containing:

1. the MPTET ID

2. the phop (u’s loopback + u’s oif for the link)

3. the allocated label L_u

u records label L_u as part of its own junction state.

When v receives a LABEL message from all its nhops, it installs swap state in its LFIB.

3.4. Forwarding State

For a non-ingress node v, forwarding state generally consists of a set of routes which identify the tunnel from its phops, and a set of weighted nexthops, i.e., a set of nexthops whereby the relative proportion of traffic sent over each is decide by the MC and specified by the SS in the JUNCTION message.

For IP tunnels, the route consists of a destination-source pair, possibly with a tunnel discriminator (which allows multiple tunnels between an ingress-egress pair); the nexthops are a set of interfaces over which the packet is forwarded.

For a MPTET with a "statically assigned" label (typically by a PCE), the route consists of the assigned label, and the nexthops are a set of interfaces. In the simplest case, the entire DAG has a single label; if so, the label operation is null. A variant allows for different controller-assigned labels for each junction node; in this case, the forwarding state is as for "signaled" labels, where the incoming label is swapped to the correct outgoing label.

For signaled labels, the routes for node v are the labels v sent to its phops. Each nexthop is a swap of the incoming label to the label sent by v’s nhops.

4. Signaling Protocol

Several signaling protocols are being extended to provision MPTETs: RSVP-TE, PCEP and BGP, among others. The details of each will be specified in companion documents; this memo restricts itself to an overview of the common elements.

4.1. Message Flow

Provisioning messages (to create, update and delete a tunnel) are sent from the Signaling Source (SS) to each junction node (including possibly other ingresses). Notifications are sent from each junction node to the SS to send updates on the state of that node with respect to the MPTET. Label messages (when needed) are sent hop-by-hop from egresses to their phops and further upstream in an ordered fashion.

In special scenarios, a node may send a messages to one or more of its nhops.

4.2. Message Types

4.2.1. JUNCTION

A JUNCTION message contains the following information elements:

MPTET ID: a unique identifier for an MPTE tunnel. This usually consists of the TO ID and a unique ID in the namespace of the TO. It also includes a version number to distinguish among instances of a tunnel as it is undergoes updates. The companion signaling documents will describe the MPTET ID in more detail.

Tunnel Type: various types of tunnels are used, so each node must be told which type of tunnel this MPTET consists of.

Tunnel Information: provides details for the MPTET. For example, for an MPLS tunnel with a statically assigned label, the Tunnel Information is the label. For IP-based tunnels, the Tunnel Information is the source and destination IP addresses (plus optional other information).

Junction Bandwidth: specifies the bandwidth incoming to the junction in Megabits per second (Mbps).

nhop share: a 2-octet share of the outgoing bandwidth per nhop. A Junction should attempt to send a ratio of (share n)/(sum (share i)) of the incoming bandwidth to nhop #n.

4.2.2. LABEL

A LABEL message MUST only be used for MPTEDs of type SigLab. A LABEL message is sent from an egress junction node to each of its phops. Any other junction node MUST only send a LABEL message when it has received a LABEL message from all of its nhops (cf "Ordered Label Distribution Control" [RFC3036], Section 2.6.1.2). A pure ingress node never sends a LABEL message as it has no phops. The LABEL message carries the MPTET ID and a label.

4.2.3. NOTIFY

A NOTIFY is sent from a junction node to the SS to let the SS know the state of the MPTET at that node. This could be the labels it assigned to its phops, or error conditions.

5. Graceful Restart

A node N is capable of Graceful Restart if a) it can maintain control plane state across restarts; and b) it can maintain forwarding state across restarts. If N is capable of Graceful Restart, an MPTE DAG going through N can continue functioning while N restarts. While N is restarting, new JUNCTION/LABEL messages will be dropped or ignored; new MPTE DAGs passing through N will not be established. Once restart is complete, N will send an OPEN message and re-establish connections will all its peers (or all the MPTEP Reflectors). Thereafter, N can participate in new DAGs passing through it by processing received JUNCTION messages.

More details will be described in a future version.

6. IANA Considerations

TBD

7. Security Considerations

TBD

8. References

8.1. Normative References

[RFC2003] Perkins, C., "IP Encapsulation within IP", RFC 2003, DOI 10.17487/RFC2003, October 1996, <https://www.rfc-editor.org/rfc/rfc2003>.

[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>.

[RFC2784] Farinacci, D., Li, T., Hanks, S., Meyer, D., and P. Traina, "Generic Routing Encapsulation (GRE)", RFC 2784, DOI 10.17487/RFC2784, March 2000, <https://www.rfc-editor.org/rfc/rfc2784>.

[RFC2890] Dommety, G., "Key and Sequence Number Extensions to GRE", RFC 2890, DOI 10.17487/RFC2890, September 2000, <https://www.rfc-editor.org/rfc/rfc2890>.

[RFC7510] Xu, X., Sheth, N., Yong, L., Callon, R., and D. Black, "Encapsulating MPLS in UDP", RFC 7510, DOI 10.17487/RFC7510, April 2015, <https://www.rfc-editor.org/rfc/rfc7510>.

[RFC8086] Yong, L., Ed., Crabbe, E., Xu, X., and T. Herbert, "GRE-in-UDP Encapsulation", RFC 8086, DOI 10.17487/RFC8086, March 2017, <https://www.rfc-editor.org/rfc/rfc8086>.

[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>.

8.2. Informative References

[RFC2702] Awduche, D., Malcolm, J., Agogbua, J., O'Dell, M., and J. McManus, "Requirements for Traffic Engineering Over MPLS", RFC 2702, DOI 10.17487/RFC2702, September 1999, <https://www.rfc-editor.org/rfc/rfc2702>.

[RFC3036] Andersson, L., Doolan, P., Feldman, N., Fredette, A., and B. Thomas, "LDP Specification", RFC 3036, DOI 10.17487/RFC3036, January 2001, <https://www.rfc-editor.org/rfc/rfc3036>.

[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>.

[RFC4090] Pan, P., Ed., Swallow, G., Ed., and A. Atlas, Ed., "Fast Reroute Extensions to RSVP-TE for LSP Tunnels", RFC 4090, DOI 10.17487/RFC4090, May 2005, <https://www.rfc-editor.org/rfc/rfc4090>.

[RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <https://www.rfc-editor.org/rfc/rfc4271>.

[RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, DOI 10.17487/RFC4364, February 2006, <https://www.rfc-editor.org/rfc/rfc4364>.

[RFC4655] Farrel, A., Vasseur, J.-P., and J. Ash, "A Path Computation Element (PCE)-Based Architecture", RFC 4655, DOI 10.17487/RFC4655, August 2006, <https://www.rfc-editor.org/rfc/rfc4655>.

[RFC4761] Kompella, K., Ed. and Y. Rekhter, Ed., "Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling", RFC 4761, DOI 10.17487/RFC4761, January 2007, <https://www.rfc-editor.org/rfc/rfc4761>.

[RFC4762] Lasserre, M., Ed. and V. Kompella, Ed., "Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signaling", RFC 4762, DOI 10.17487/RFC4762, January 2007, <https://www.rfc-editor.org/rfc/rfc4762>.

[RFC7432] Sajassi, A., Ed., Aggarwal, R., Bitar, N., Isaac, A., Uttaro, J., Drake, J., and W. Henderickx, "BGP MPLS-Based Ethernet VPN", RFC 7432, DOI 10.17487/RFC7432, February 2015, <https://www.rfc-editor.org/rfc/rfc7432>.

[SPF] Dijkstra, E. W., "A note on two problems in connexion with graphs", 1 December 1959, <https://doi.org/10.1007/BF01386390>.

Authors' Addresses

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

Luay Jalil
Verizon
Richardson, Texas 75081
United States of America
Email: luay.jalil@verizon.com

Mazen Khaddam
Cox Communications
Atlanta, Georgia 30328
United States of America
Email: mazen.khaddam@cox.com

Andy Smith
Oracle Cloud Infrastructure
Austin, Texas 78741
United States of America
Email: andy.j.smith@oracle.com