Internet Engineering Task Force (IETF)
Request for Comments: 6790
Updates: 3031, 3107, 3209, 5036
Category: Standards Track
ISSN: 2070-1721

K. Kompella
J. Drake
Juniper Networks
S. Amante
Level 3 Communications, Inc.
W. Henderickx
Alcatel-Lucent
L. Yong
Huawei USA

November 2012

The Use of Entropy Labels in MPLS Forwarding

Abstract

Load balancing is a powerful tool for engineering traffic across a network. This memo suggests ways of improving load balancing across MPLS networks using the concept of "entropy labels". It defines the concept, describes why entropy labels are useful, enumerates properties of entropy labels that allow maximal benefit, and shows how they can be signaled and used for various applications. This document updates RFCs 3031, 3107, 3209, and 5036.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc6790.

Table of Contents

1. Introduction

Load balancing, or multi-pathing, is an attempt to balance traffic across a network by allowing the traffic to use multiple paths. Load balancing has several benefits: it eases capacity planning, it can help absorb traffic surges by spreading them across multiple paths, and it allows better resilience by offering alternate paths in the event of a link or node failure.

As providers scale their networks, they use several techniques to achieve greater bandwidth between nodes. Two widely used techniques are: Link Aggregation Group (LAG) and Equal Cost Multi-Path (ECMP). LAG is used to bond together several physical circuits between two adjacent nodes so they appear to higher-layer protocols as a single, higher-bandwidth "virtual" pipe. ECMP is used between two nodes separated by one or more hops, to allow load balancing over several shortest paths in the network. This is typically obtained by arranging IGP metrics such that there are several equal cost paths between source-destination pairs. Both of these techniques may, and often do, coexist in differing parts of a given provider's network, depending on various choices made by the provider.

A very important requirement when load balancing is that packets belonging to a given "flow" must be mapped to the same path, i.e., the same exact sequence of links across the network. This is to avoid jitter, latency, and reordering issues for the flow. What constitutes a flow varies considerably. A common example of a flow is a TCP session. Other examples are a Layer 2 Tunneling Protocol (L2TP) session corresponding to a given broadband user or traffic within an ATM virtual circuit.

To meet this requirement, a node uses certain fields, termed "keys", within a packet's header as input to a load-balancing function (typically a hash function) that selects the path for all packets in a given flow. The keys chosen for the load-balancing function depend on the packet type; a typical set (for IP packets) is the IP source and destination addresses, the protocol type, and (for TCP and UDP traffic) the source and destination port numbers. An overly conservative choice of fields may lead to many flows mapping to the same hash value (and consequently poorer load balancing); an overly aggressive choice may map a flow to multiple values, potentially violating the above requirement.

For MPLS networks, most of the same principles (and benefits) apply. However, finding useful keys in a packet for the purpose of load balancing can be more of a challenge. In many cases, MPLS encapsulation may require fairly deep inspection of packets to find these keys at transit Label Switching Routers (LSRs).

One way to eliminate the need for this deep inspection is to have the ingress LSR of an MPLS Label Switched Path extract the appropriate keys from a given packet, input them to its load-balancing function, and place the result in an additional label, termed the "entropy label", as part of the MPLS label stack it pushes onto that packet.

The entire label stack of the MPLS packet can then be used by transit LSRs to perform load balancing, as the entropy label introduces the right level of "entropy" into the label stack.

There are five key reasons why this is beneficial:

1. At the ingress LSR, MPLS encapsulation hasn't yet occurred, so deep inspection is not necessary.

2. The ingress LSR has more context and information about incoming packets than transit LSRs.

3. Ingress LSRs usually operate at lower bandwidths than transit LSRs, allowing them to do more work per packet.

4. Transit LSRs do not need to perform deep packet inspection and can load balance effectively using only a packet's MPLS label stack.

5. Transit LSRs, not having the full context that an ingress LSR does, have the hard choice between potentially misinterpreting fields in a packet as valid keys for load balancing (causing packet-ordering problems) or adopting a conservative approach (giving rise to sub-optimal load balancing). Entropy labels relieve them of making this choice.

This memo describes why entropy labels are needed and defines the properties of entropy labels, in particular, how they are generated and received and the expected behavior of transit LSRs. Finally, it describes in general how signaling works and what needs to be signaled as well as specifics for the signaling of entropy labels for LDP [RFC5036], BGP [RFC4271], and RSVP - Traffic Engineering (RSVP-TE) [RFC3209].

1.1. Conventions Used

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

The following acronyms/initialisms are used:

BoS: Bottom of Stack

CE: Customer Edge

ECMP: Equal Cost Multi-Path

EL: Entropy Label

ELC: Entropy Label Capability

ELI: Entropy Label Indicator

FEC: Forwarding Equivalence Class

LAG: Link Aggregation Group

LER: Label Edge Router

LSP: Label Switched Path

LSR: Label Switching Router

PE: Provider Edge

PW: Pseudowire

PHP: Penultimate Hop Popping

TC: Traffic Class

TTL: Time to Live

UHP: Ultimate Hop Popping

VPLS: Virtual Private LAN (Local Area Network) Service

VPN: Virtual Private Network

The term "ingress LSR" (or "egress LSR") is used interchangeably with "ingress LER" (or "egress LER"). The term "application" throughout the text refers to an MPLS application (such as a VPN or VPLS).

A label stack (say of three labels) is denoted by <L1, L2, L3>, where L1 is the "outermost" label and L3 the "innermost" (closest to the payload). Packet flows are depicted left to right, and signaling is shown right to left (unless otherwise indicated).

The term "label" is used both for the entire 32-bit label stack entry and the 20-bit label field within a label stack entry. It should be clear from the context which is meant.

1.2. Motivation

MPLS is a very successful generic forwarding substrate that transports several dozen types of protocols, most notably: IP, PWs, VPLS, and IP VPNs. Within each type of protocol, there typically exist several variants, each with a different set of load-balancing keys, e.g., IPv4, IPv6, IPv6 in IPv4, etc. for IP and Ethernet; ATM, Frame-Relay, etc. for PWs. There are also several different types of Ethernet over PW encapsulation, ATM over PW encapsulation, etc. Finally, given the popularity of MPLS, it is likely that it will continue to be extended to transport new protocols.

Currently, each transit LSR along the path of a given LSP has to try to infer the underlying protocol within an MPLS packet in order to extract appropriate keys for load balancing. Unfortunately, if the transit LSR is unable to infer the MPLS packet's protocol (as is often the case), it will typically use the topmost (or all) MPLS labels in the label stack as keys for the load-balancing function. The result may be an extremely inequitable distribution of traffic across equal cost paths exiting that LSR. This is because MPLS labels are generally fairly coarse-grained forwarding labels that typically describe a next hop, or provide some demultiplexing and/or forwarding function, and do not describe the packet's underlying protocol.

On the other hand, an ingress LSR (e.g., a PE router) has detailed knowledge of a packet's contents, typically through a priori configuration of the encapsulations that are expected at a given PE-CE interface, (e.g., IPv4, IPv6, VPLS, etc.). They may also have more flexible forwarding hardware, depending on implementation details. PE routers need this information and these capabilities to:

a. apply the required services for the CE;

b. discern the packet's Class of Service (CoS) forwarding treatment;

c. apply filters to forward or block traffic to/from the CE;

d. forward routing/control traffic to an onboard management processor; and,

e. load balance the traffic on its uplinks to transit LSRs (e.g., P routers).

By knowing the expected encapsulation types, an ingress LSR router can apply a more specific set of payload parsing routines to extract the keys appropriate for a given protocol. This allows for significantly improved accuracy in determining the appropriate load-balancing behavior for each protocol.

If the ingress LSR were to capture the flow information so gathered in a convenient form for downstream transit LSRs, transit LSRs could remain completely oblivious to the contents of each MPLS packet and use only the captured flow information to perform load balancing. In particular, there will be no reason to duplicate an ingress LSR's complex packet/payload parsing functionality in a transit LSR. This will result in less complex transit LSRs, enabling them to more easily scale to higher forwarding rates, larger port density, lower power consumption, etc. The idea in this memo is to capture this flow information as a label, the so-called "entropy label".

Ingress LSRs can also adapt more readily to new protocols and extract the appropriate keys to use for load-balancing packets of those protocols. This means that deploying new protocols or services in edge devices requires fewer concomitant changes in the core, resulting in higher edge-service velocity and, at the same time, more stable core networks.

2. Approaches

There are two main approaches to encoding load-balancing information in the label stack. The first allocates multiple labels for a particular Forwarding Equivalence Class (FEC). These labels are equivalent in terms of forwarding semantics, but having multiple labels allows flexibility in assigning labels to flows belonging to the same FEC. This approach has the advantage that the label stack has the same depth whether or not one uses label-based load balancing; consequently, there is no change to forwarding operations on transit and egress LSRs. However, it has a major drawback in that there is a significant increase in both signaling and forwarding state.

The other approach encodes the load-balancing information as an additional label in the label stack, thus increasing the depth of the label stack by one. With this approach, there is minimal change to signaling state for a FEC; also, there is no change in forwarding operations in transit LSRs and no increase of forwarding state in any LSR. The only purpose of the additional label is to increase the entropy in the label stack, so this is called an "entropy label". This memo focuses solely on this approach.

This latter approach uses upstream-generated entropy labels, which may conflict with downstream-allocated application labels. There are a few approaches to deal with this: 1) allocate a pair of labels for each FEC, one that must have an entropy label below it and one that must not; 2) use a label (the "Entropy Label Indicator") to indicate that the next label is an entropy label; and 3) allow entropy labels only where there is no possible confusion. The first doubles control and data plane state in the network; the last is too restrictive. The approach taken here is the second. In making both the above choices, the trade-off is to increase label stack depth rather than control and data plane state in the network.

Finally, one may choose to associate ELs with MPLS tunnels (LSPs) or with MPLS applications (e.g., VPNs). (What this entails is described in later sections.) We take the former approach, for the following reasons:

1. There are a small number of tunneling protocols for MPLS, but a large and growing number of applications. Defining ELs on a tunnel basis means simpler standards, lower development, interoperability, and testing efforts.

2. As a consequence, there will be much less churn in the network as new applications (services) are defined and deployed.

3. Processing application labels in the data plane is more complex than processing tunnel labels. Thus, it is preferable to burden the latter rather than the former with EL processing.

4. Associating ELs with tunnels makes it simpler to deal with hierarchy, be it LDP-over-RSVP-TE or Carrier's Carrier VPNs. Each layer in the hierarchy can choose independently whether or not they want ELs.

The cost of this approach is that ELIs will be mandatory; again, the trade-off is the size of the label stack. To summarize, the net increase in the label stack to use entropy labels is two: one reserved label for the ELI and the entropy label itself.

3. Entropy Labels and Their Structure

An entropy label (as used here) is a label:

1. that is not used for forwarding;

2. that is not signaled; and

3. whose only purpose in the label stack is to provide "entropy" to improve load balancing.

Entropy labels are generated by an ingress LSR, based entirely on load-balancing information. However, they MUST NOT have values in the reserved label space (0-15) [RFC3032].

Since entropy labels are generated by an ingress LSR, an egress LSR MUST be able to distinguish unambiguously between entropy labels and application labels. To accomplish this, it is REQUIRED that the label immediately preceding an Entropy Label (EL) in the MPLS label stack be an Entropy Label Indicator (ELI), where preceding means closer to the top of the label stack (farther from bottom of stack indication). The ELI is a reserved label with value 7. How to set values of the TTL, TC, and "Bottom of Stack" (BoS) fields [RFC3032] for the ELI and for ELs is discussed in Section 4.2.

Entropy labels are useful for pseudowires [RFC4447]. [RFC6391] explains how entropy labels can be used for pseudowires that are of the RFC 4447 style and is therefore complementary to this memo, which focuses on how entropy labels can be used for tunnels and thus for all other MPLS applications.

4. Data Plane Processing of Entropy Labels

4.1. Egress LSR

Suppose egress LSR Y is capable of processing entropy labels for a tunnel. Y indicates this to all ingresses via signaling (see Section 5). Y MUST be prepared to deal both with packets with an imposed EL and those without; the ELI will distinguish these cases. If a particular ingress chooses not to impose an EL, Y's processing of the received label stack (which might be empty) is as if Y chose not to accept ELs.

If an ingress LSR X chooses to impose an EL, then Y will receive a tunnel termination packet with label stack <TL, ELI, EL> <remaining packet header>. Y recognizes TL as the label it distributed to its upstreams for the tunnel and pops it. (Note that TL may be the implicit null label, in which case it doesn't appear in the label stack.) Y then recognizes the ELI and pops two labels: the ELI and the EL. Y then processes the remaining packet header as normal; this may require further processing of tunnel termination, perhaps with further ELI+EL pairs. When processing the final tunnel termination, Y MAY enqueue the packet based on that tunnel TL's or ELI's TC value and MAY use the tunnel TL's or ELI's TTL to compute the TTL of the remaining packet header. The EL's TTL MUST be ignored.

If any ELI processed by Y has the BoS bit set, Y MUST discard the packet and MAY log an error. The EL's BoS bit will indicate whether or not there are more labels in the stack.

4.2. Ingress LSR

If an egress LSR Y indicates via signaling that it can process ELs on a particular tunnel, an ingress LSR X can choose whether or not to insert ELs for packets going into that tunnel. Y MUST handle both cases.

The steps that X performs to insert ELs are as follows:

1. On an incoming packet, identify the application to which the packet belongs; based on this, pick appropriate fields as input to the load-balancing function; apply the load-balancing function to these input fields and let LB be the output.

2. Determine the application label AL (if any). Push <AL> onto the packet.

3. Based on the application, the load-balancing output LB and other factors, determine the egress LSR Y, the tunnel to Y, the specific interface to the next hop, and thus the tunnel label TL. Use LB to generate the entropy label EL.

4. If, for the chosen tunnel, Y has not indicated that it can process ELs, push <TL> onto the packet. If Y has indicated that it can process ELs for the tunnel, push <TL, ELI, EL> onto the packet. X SHOULD put the same TTL and TC fields for the ELI as it does for TL. X MAY choose different values for the TTL and TC fields if it is known that the ELI will not be exposed as the top label at any point along the LSP (as may happen in cases where PHP is used and the ELI and EL are not stripped at the penultimate hop (see Section 4.4). The BoS bit for the ELI MUST be zero (i.e., BoS is not set). The TTL for the EL MUST be zero to ensure that it is not used inadvertently for forwarding. The TC for the EL may be any value. The BoS bit for the EL depends on whether or not there are more labels in the label stack.

5. X then determines whether further tunnel hierarchy is needed; if so, X goes back to step 3, possibly with a new egress Y for the new tunnel. Otherwise, X is done and sends out the packet.

Notes:

a. X computes load-balancing information and generates the EL based on the incoming application packet, even though the signaling of EL capability is associated with tunnels.

b. X MAY insert several entropy labels in the stack (each, of course, preceded by an ELI), potentially one for each hierarchical tunnel, provided that the egress for that tunnel has indicated that it can process ELs for that tunnel.

c. X MUST NOT include an entropy label for a given tunnel unless the egress LSR Y has indicated that it can process entropy labels for that tunnel.

d. The signaling and use of entropy labels in one direction (signaling from Y to X and data path from X to Y) is completely independent of the signaling and use of entropy labels in the reverse direction (signaling from X to Y and data path from Y to X).

4.3. Transit LSR

Transit LSRs MAY operate with no change in forwarding behavior. The following are suggestions for optimizations that improve load balancing, reduce the amount of packet data processed, and/or enhance backward compatibility.

If a transit LSR recognizes the ELI, it MAY choose to load balance solely on the following label (the EL); otherwise, it SHOULD use as much of the whole label stack as feasible as keys for the load-balancing function. In any case, reserved labels MUST NOT be used as keys for the load-balancing function.

Some transit LSRs look beyond the label stack for better load-balancing information. This is a simple, backward-compatible approach in networks where some ingress LSRs impose ELs and others don't. However, this is of limited incremental value if an EL is indeed present and requires more packet processing from the LSR. A transit LSR MAY choose to parse the label stack for the presence of the ELI and look beyond the label stack only if it does not find it, thus retaining the old behavior when needed, yet avoiding unnecessary work if not needed.

As stated in Sections 4.1 and 5, an egress LSR that signals both ELC and implicit null MUST pop the ELI and the next label (which should be the EL), if it encounters a packet with the ELI as the topmost label. Any other LSR (including PHP LSRs) MUST drop such packets, as per Section 3.18 of [RFC3031].

4.4. Penultimate Hop LSR

No change is needed at penultimate hop LSRs. However, a PHP LSR that recognizes the ELI MAY choose to pop the ELI and following label (which should be an entropy label) in addition to popping the tunnel label, provided that doing so doesn't diminish its ability to load balance on the next hop.

5. Signaling for Entropy Labels

An egress LSR Y can signal to ingress LSR(s) its ability to process entropy labels (henceforth called "Entropy Label Capability" or ELC) on a given tunnel. In particular, even if Y signals an implicit null label, indicating that PHP is to be performed, Y MUST be prepared to pop the ELI and EL.

Note that Entropy Label Capability may be asymmetric: if LSRs X and Y are at opposite ends of a tunnel, X may be able to process entropy labels, whereas Y may not. The signaling extensions below allow for this asymmetry.

For an illustration of signaling and forwarding with entropy labels, see Section 8.

5.1. LDP Signaling

A new LDP TLV [RFC5036] is defined to signal an egress's ability to process entropy labels. This is called the ELC TLV and may appear as an Optional Parameter of the Label Mapping Message TLV.

The presence of the ELC TLV in a Label Mapping Message indicates to ingress LSRs that the egress LSR can process entropy labels for the associated LDP tunnel. The ELC TLV has Type 0x0206 and Length 0.

The structure of the ELC TLV is shown below.

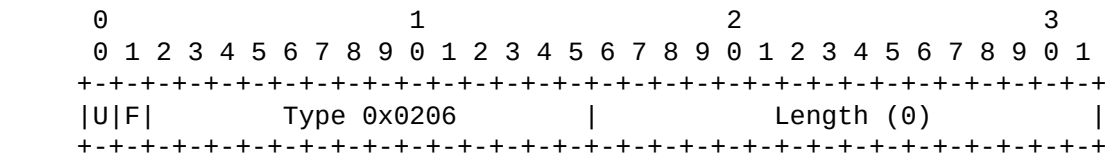

Figure 1: Entropy Label Capability TLV

where:

U: Unknown bit. This bit MUST be set to 1. If the ELC TLV is not understood by the receiver, then it MUST be ignored.

F: Forward bit. This bit MUST be set be set to 1. Since the ELC TLV is going to be propagated hop-by-hop, it should be forwarded even by nodes that may not understand it.

Type: Type field (0x0206)

Length: Length field. This field specifies the total length in octets of the ELC TLV and is currently defined to be 0.

5.1.1. Processing the ELC TLV

An LSR that receives a Label Mapping with the ELC TLV but does not understand it MUST propagate it intact to its neighbors and MUST NOT send a notification to the sender (following the meaning of the U- and F-bits).

An LSR X may receive multiple Label Mappings for a given FEC F from its neighbors. In its turn, X may advertise a Label Mapping for F to its neighbors. If X understands the ELC TLV, and if any of the advertisements it received for FEC F does not include the ELC TLV, X MUST NOT include the ELC TLV in its own advertisements of F. If all the advertised Mappings for F include the ELC TLV, then X MUST advertise its Mapping for F with the ELC TLV. If any of X's neighbors resends its Mapping, sends a new Mapping or sends a Label Withdraw for a previously advertised Mapping for F, X MUST re-evaluate the status of ELC for FEC F, and, if there is a change, X MUST re-advertise its Mapping for F with the updated status of ELC.

5.2. BGP Signaling

When BGP [RFC4271] is used for distributing Network Layer Reachability Information (NLRI) as described in, for example, [RFC3107], the BGP UPDATE message may include the ELC attribute as part of the Path Attributes. This is an optional, transitive BGP

attribute of value 28. The inclusion of this attribute with an NLRI indicates that the advertising BGP router can process entropy labels as an egress LSR for all routes in that NLRI.

A BGP speaker S that originates an UPDATE should include the ELC attribute only if both of the following are true:

A1: S sets the BGP NEXT_HOP attribute to itself AND

A2: S can process entropy labels.

Suppose a BGP speaker T receives an UPDATE U with the ELC attribute. T has two choices. T can simply re-advertise U with the ELC attribute if either of the following is true:

B1: T does not change the NEXT_HOP attribute OR

B2: T simply swaps labels without popping the entire label stack and processing the payload below.

An example of the use of B1 is Route Reflectors.

However, if T changes the NEXT_HOP attribute for U and in the data plane pops the entire label stack to process the payload, T MAY include an ELC attribute for UPDATE U' if both of the following are true:

C1: T sets the NEXT_HOP attribute of U' to itself AND

C2: T can process entropy labels.

Otherwise, T MUST remove the ELC attribute.

5.3. RSVP-TE Signaling

Entropy label support is signaled in RSVP-TE [RFC3209] using the Entropy Label Capability (ELC) flag in the Attribute Flags TLV of the LSP_ATTRIBUTES object [RFC5420]. The presence of the ELC flag in a Path message indicates that the ingress can process entropy labels in the upstream direction; this only makes sense for a bidirectional LSP and MUST be ignored otherwise. The presence of the ELC flag in a Resv message indicates that the egress can process entropy labels in the downstream direction.

The bit number for the ELC flag is 9.

5.4. Multicast LSPs and Entropy Labels

Multicast LSPs [RFC4875] [RFC6388] typically do not use ECMP for load balancing, as the combination of replication and multi-pathing can lead to duplicate traffic delivery. However, these LSPs can traverse bundled links [RFC4201] and LAGs. In both these cases, load balancing is useful, and hence entropy labels can be of value for multicast LSPs.

The methodology defined for entropy labels here will be used for multicast LSPs; however, the details of signaling and processing ELs for multicast LSPs will be specified in a future document.

6. Operations, Administration, and Maintenance (OAM) and Entropy Labels

Generally, OAM comprises a set of functions operating in the data plane to allow a network operator to monitor its network infrastructure and to implement mechanisms in order to enhance the general behavior and the level of performance of its network, e.g., the efficient and automatic detection, localization, diagnosis, and handling of defects.

Currently defined OAM mechanisms for MPLS include LSP ping/traceroute [RFC4379] and Bidirectional Forwarding Detection (BFD) for MPLS [RFC5884]. The latter provides connectivity verification between the endpoints of an LSP, and recommends establishing a separate BFD session for every path between the endpoints.

The LSP traceroute procedures of [RFC4379] allow an ingress LSR to obtain label ranges that can be used to send packets on every path to the egress LSR. It works by having the ingress LSR sequentially ask the transit LSRs along a particular path to a given egress LSR to return a label range such that the inclusion of a label in that range in a packet will cause the replying transit LSR to send that packet out the egress interface for that path. The ingress provides the label range returned by transit LSR N to transit LSR N + 1, which returns a label range that is less than or equal in span to the range provided to it. This process iterates until the penultimate transit LSR replies to the ingress LSR with a label range that is acceptable to it and to all LSRs along path preceding it for forwarding a packet along the path.

However, the LSP traceroute procedures do not specify where in the label stack the value from the label range is to be placed, whether deep packet inspection is allowed, and if so, which keys and key values are to be used.

This memo updates LSP traceroute by specifying that the value from the label range is to be placed in the entropy label. Deep packet inspection is thus not necessary, although an LSR may use it, provided it does so consistently, i.e., if the label range to go to a given downstream LSR is computed with deep packet inspection, then the data path should use the same approach and the same keys.

In order to have a BFD session on a given path, a value from the label range for that path should be used as the EL value for BFD packets sent on that path.

7. MPLS-TP and Entropy Labels

Since the MPLS Transport Profile (MPLS-TP) does not use ECMP, entropy labels are not applicable to an MPLS-TP deployment.

8. Entropy Labels in Various Scenarios

This section describes the use of entropy labels in various scenarios. The material in this section is illustrative and offers guidance to implementations, but it does not form a normative part of this specification.

In the figures below, the following conventions are used to depict processing between X and Y. Note that control plane signaling goes right to left, whereas data plane processing goes left to right.

```
Protocols
Y:        <--- [L, E]                        Y signals L to X
    X ------------- Y
Data Plane:
X-Y:  <L, ELI, EL>                           Label Stack from X -> Y
Label Stack Operations:
X:  +<L, ELI, EL>                            X pushes <L, ELI, EL>
Y:                  -<L, ELI, EL>            Y pops <L, ELI, EL>
```

This means that Y signals to X label L for an LDP tunnel. E can be one of:

0: meaning egress is NOT entropy label capable or

1: meaning egress is entropy label capable

The line with LS: shows the label stack on the wire. Below that is the operation that each LSR does in the data plane, where + means push the following label stack, - means pop the following label stack, L~L’ means swap L with L’.

8.1. LDP Tunnel

The following figures illustrate several simple intra-AS LDP tunnels. The first diagram shows ultimate hop popping (UHP) with the ingress inserting an EL, the second UHP with no ELs, the third PHP with ELs, and finally, PHP with no ELs, but also with an application label AL (which could, for example, be a VPN label).

Note that, in all the cases below, the MPLS application does not matter; it may be that X pushes some more labels (perhaps for a VPN or VPLS) below the ones shown, and Y pops them.

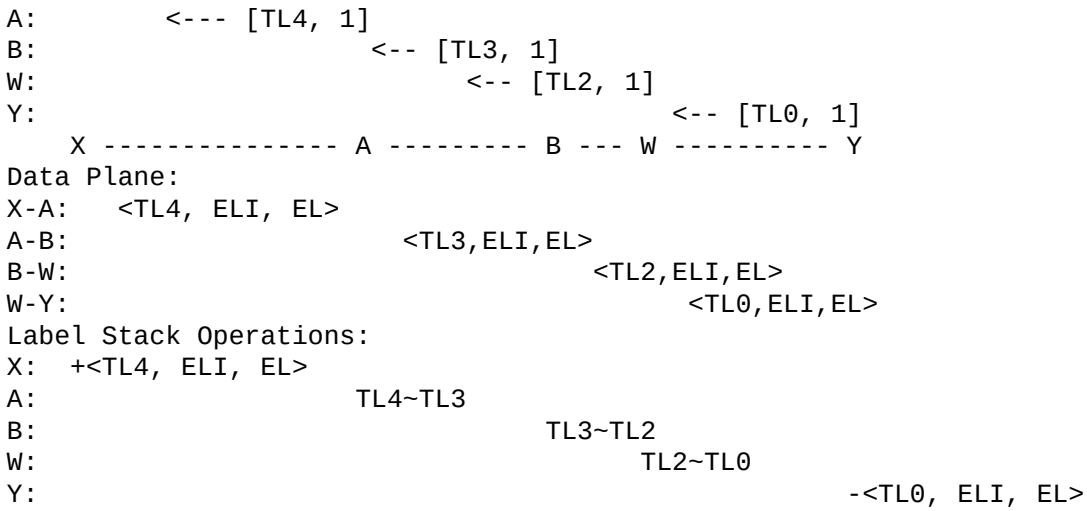

Figure 2: LDP with UHP; Ingress Inserts ELs

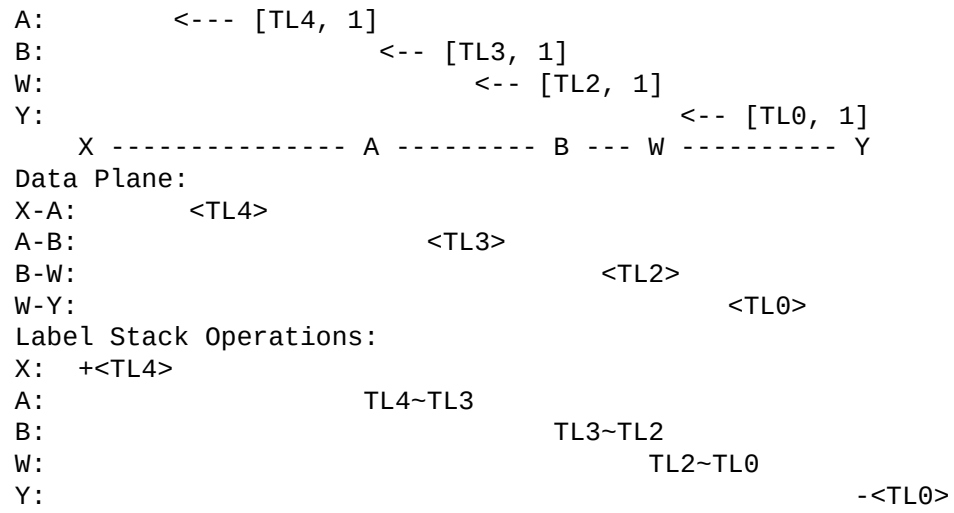

Figure 3: LDP with UHP; Ingress Does Not Insert ELs

Note that in Figure 3, above, the Egress Y is signaling it is EL-capable, but the Ingress X has chosen not to insert ELs.

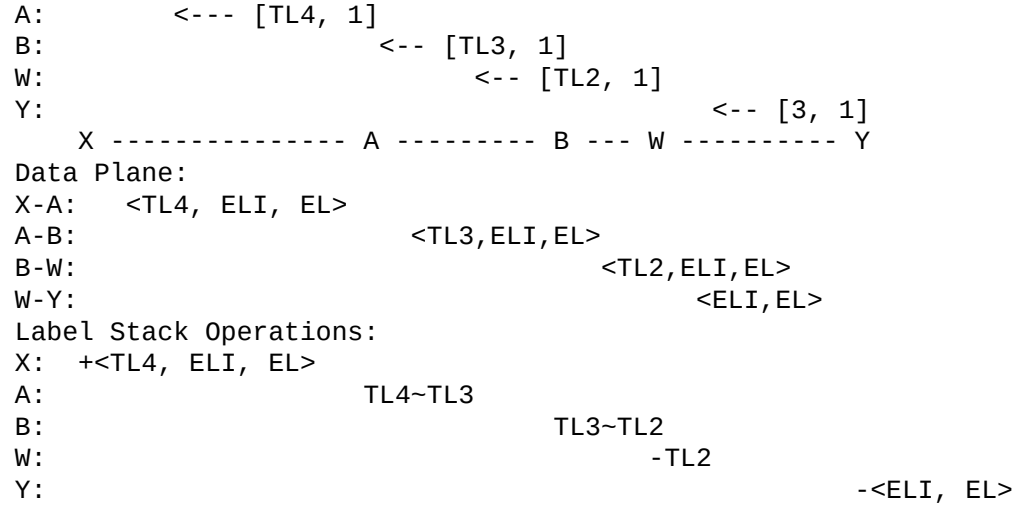

Figure 4: LDP with PHP; Ingress Inserts ELs

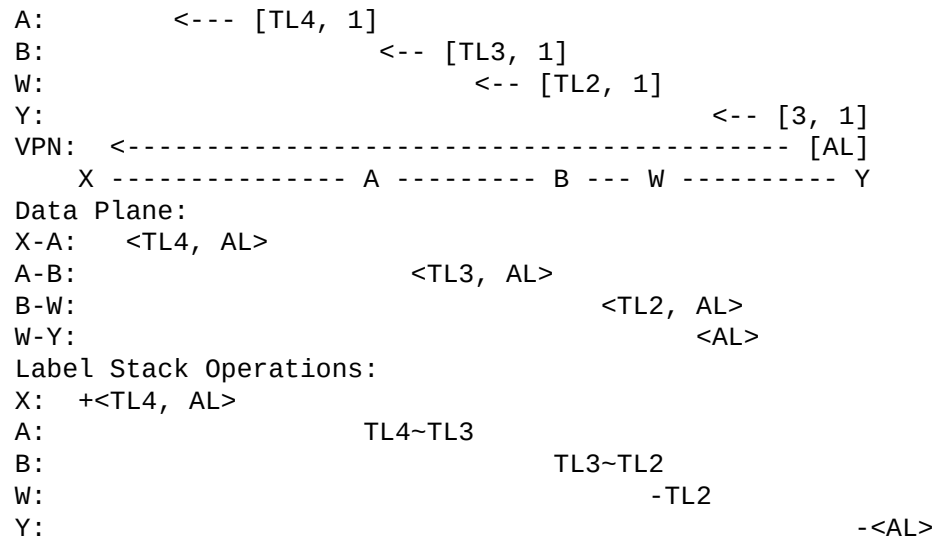

Figure 5: LDP with PHP + VPN; Ingress Does Not Insert ELs

Note that in Figure 5, above, the Egress Y is signaling it is EL-capable, but the Ingress X has chosen not to insert ELs.

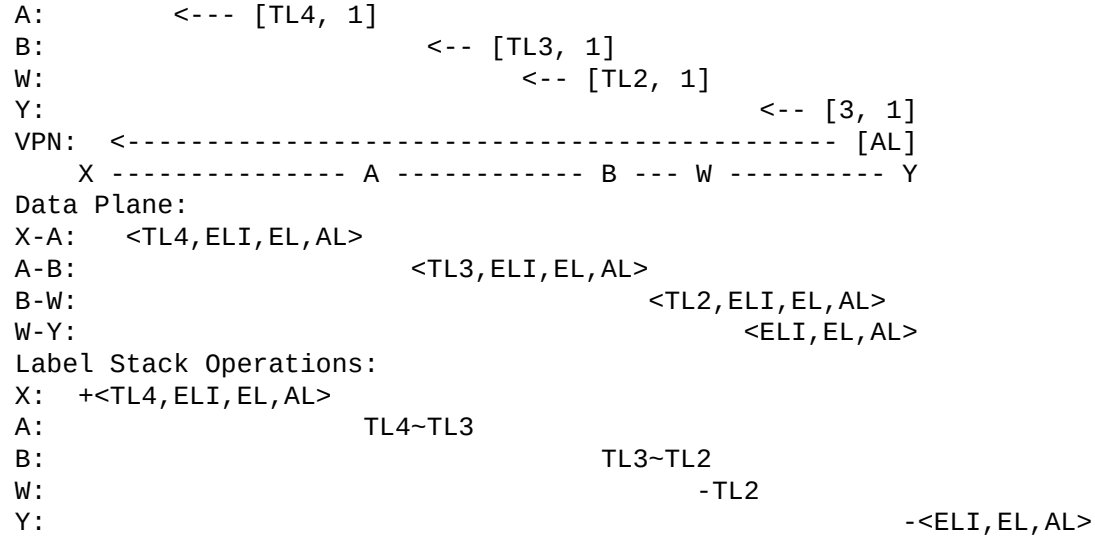

Figure 6: LDP with PHP + VPN; Ingress Inserts ELs

8.2. LDP over RSVP-TE

Figure 7 illustrates "LDP over RSVP-TE" tunnels. X and Y are the ingress and egress (respectively) of the LDP tunnel; A and W are the ingress and egress of the RSVP-TE tunnel. It is assumed that both the LDP and RSVP-TE tunnels have PHP.

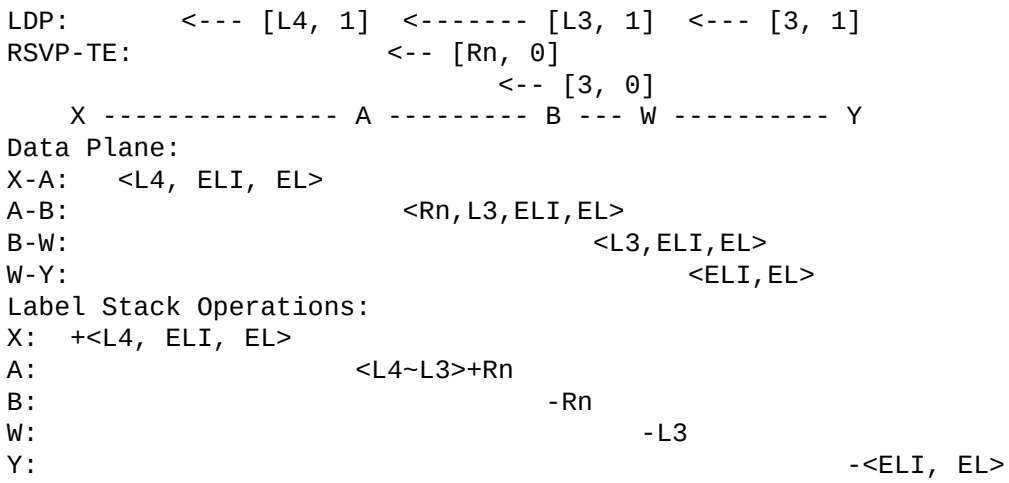

Figure 7: LDP with ELs over RSVP-TE Tunnels without ELs

8.3. MPLS Applications

For each unicast tunnel starting at an ingress LSR X, X must remember whether the egress for that tunnel can process entropy labels. X does not have to keep state per application running over that tunnel. However, an ingress PE can choose on a per-application basis whether or not to insert ELs. For example, X may have an application for which it does not wish to use ECMP (e.g., circuit emulation) or for which it does not know which keys to use for load balancing (e.g., Appletalk over a pseudowire). In either of those cases, X may choose not to insert entropy labels but may choose to insert entropy labels for an IP VPN over the same tunnel.

9. Security Considerations

This document describes advertisement of the capability to support receipt of entropy labels that an ingress LSR may insert in MPLS packets in order to allow transit LSRs to attain better load balancing across LAG and/or ECMP paths in the network.

This document does not introduce new security vulnerabilities to LDP, BGP or RSVP-TE. Please refer to the Security Considerations sections of these protocols ([RFC5036], [RFC4271], and [RFC3209]) for security mechanisms applicable to each.

Given that there is no end-user control over the values used for entropy labels, there is little risk of entropy label forgery, which could cause uneven load balancing in the network. Note that if the EL value is calculated only based on packet headers, then a relatively efficient wiretapping interface could be added depending on the function used to generate the EL value. An implementation may protect against this by adding some other input to the generation of the EL values that would make it harder to build a table of EL values to tap given knowledge of the keys from the packet. For example, the ingress LSR could generate a random input to the EL generation process. In practice, many ECMP hashing algorithms contain a random factor in any case so as to avoid polarization issues.

If Entropy Label Capability is not signaled from an egress PE to an ingress PE, due to, for example, malicious configuration activity on the egress PE, then the PE will fall back to not using entropy labels for load balancing traffic over LAG or ECMP paths, which is, in general, no worse than the behavior observed in current production networks. That said, it is recommended that operators monitor changes to PE configurations and, more importantly, the fairness of load distribution over LAG or ECMP paths. If the fairness of load distribution over a set of paths changes that could indicate a misconfiguration, bug, or other non-optimal behavior on their PEs, and they should take corrective action.

10. IANA Considerations

10.1. Reserved Label for ELI

IANA has allocated a reserved label for the Entropy Label Indicator (ELI) from the "Multiprotocol Label Switching Architecture (MPLS) Label Values" registry.

10.2. LDP Entropy Label Capability TLV

IANA has allocated the value of 0x0206 from the IETF Consensus range (0x0001-0x07FF) in the "TLV Type Name Space" registry as the "Entropy Label Capability TLV".

10.3. BGP Entropy Label Capability Attribute

IANA has allocated the Path Attribute Type Code 28 from the "BGP Path Attributes" registry as the "BGP Entropy Label Capability Attribute".

10.4. RSVP-TE Entropy Label Capability Flag

IANA has allocated a new bit from the "Attribute Flags" sub-registry of the "Resource Reservation Protocol-Traffic Engineering (RSVP-TE) Parameters" registry.

Bit No	Name	Attribute Flags Path	Attribute Flags Resv	RRO
9	Entropy Label Capability	Yes	Yes	No

11. Acknowledgments

We wish to thank Ulrich Drafz for his contributions, as well as the entire "hash label" team for their valuable comments and discussion.

Sincere thanks to Nischal Sheth for his many suggestions and comments and for his careful reading of the document, especially with regard to data plane processing of entropy labels.

Most of the work Kireeti Kompella did on this document was done while he was at Juniper Networks. He has since moved to Contrail Systems.

12. References

12.1. Normative References

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

[RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, January 2001.

[RFC3032] Rosen, E., Tappan, D., Fedorkow, G., Rekhter, Y., Farinacci, D., Li, T., and A. Conta, "MPLS Label Stack Encoding", RFC 3032, January 2001.

[RFC3107] Rekhter, Y. and E. Rosen, "Carrying Label Information in BGP-4", RFC 3107, May 2001.

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, December 2001.

[RFC5036] Andersson, L., Minei, I., and B. Thomas, "LDP Specification", RFC 5036, October 2007.

[RFC5420] Farrel, A., Papadimitriou, D., Vasseur, JP., and A. Ayyangarps, "Encoding of Attributes for MPLS LSP Establishment Using Resource Reservation Protocol Traffic Engineering (RSVP-TE)", RFC 5420, February 2009.

12.2. Informative References

[RFC4201] Kompella, K., Rekhter, Y., and L. Berger, "Link Bundling in MPLS Traffic Engineering (TE)", RFC 4201, October 2005.

[RFC4271] Rekhter, Y., Li, T., and S. Hares, "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, January 2006.

[RFC4379] Kompella, K. and G. Swallow, "Detecting Multi-Protocol Label Switched (MPLS) Data Plane Failures", RFC 4379, February 2006.

[RFC4447] Martini, L., Rosen, E., El-Aawar, N., Smith, T., and G. Heron, "Pseudowire Setup and Maintenance Using the Label Distribution Protocol (LDP)", RFC 4447, April 2006.

[RFC4875] Aggarwal, R., Papadimitriou, D., and S. Yasukawa, "Extensions to Resource Reservation Protocol - Traffic Engineering (RSVP-TE) for Point-to-Multipoint TE Label Switched Paths (LSPs)", RFC 4875, May 2007.

[RFC5884] Aggarwal, R., Kompella, K., Nadeau, T., and G. Swallow, "Bidirectional Forwarding Detection (BFD) for MPLS Label Switched Paths (LSPs)", RFC 5884, June 2010.

[RFC6388] Wijnands, IJ., Minei, I., Kompella, K., and B. Thomas, "Label Distribution Protocol Extensions for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths", RFC 6388, November 2011.

[RFC6391] Bryant, S., Filsfils, C., Drafz, U., Kompella, V., Regan, J., and S. Amante, "Flow-Aware Transport of Pseudowires over an MPLS Packet Switched Network", RFC 6391, November 2011.

Appendix A. Applicability of LDP Entropy Label Capability TLV

In the case of unlabeled IPv4 (Internet) traffic, the best practice is for an egress LSR to propagate eBGP learned routes within a Service Provider's Autonomous System after resetting the BGP next-hop attribute to one of its loopback IP addresses. That loopback IP address is injected into the Service Provider's IGP and, concurrently, a label assigned to it via LDP. Thus, when an ingress LSR is performing a forwarding lookup for a BGP destination, it recursively resolves the associated next hop to a loopback IP address and associated LDP label of the egress LSR.

Thus, in the context of unlabeled IPv4 traffic, the LDP Entropy Label Capability TLV will typically be applied only to the FEC for the loopback IP address of the egress LSR, and the egress LSR need not announce an Entropy Label Capability for the eBGP learned route.

Authors' Addresses

Kireeti Kompella
Contrail Systems
2350 Mission College Blvd.
Santa Clara, CA 95054
US

EMail: kireeti.kompella@gmail.com

John Drake
Juniper Networks
1194 N. Mathilda Ave.
Sunnyvale, CA 94089
US

EMail: jdrake@juniper.net

Shane Amante
Level 3 Communications, Inc.
1025 Eldorado Blvd
Broomfield, CO 80021
US

EMail: shane@level3.net

Wim Henderickx
Alcatel-Lucent
Copernicuslaan 50
2018 Antwerp
Belgium

EMail: wim.henderickx@alcatel-lucent.com

Lucy Yong
Huawei USA
5340 Legacy Dr.
Plano, TX 75024
US

EMail: lucy.yong@huawei.com