Network Working Group
Internet Draft
Expiration Date: December 1997

Eric C. Rosen
Yakov Rekhter
Daniel Tappan
Dino Farinacci
Guy Fedorkow
Cisco Systems, Inc.

Tony Li
Juniper Networks, Inc.

June 1997

Label Switching: Label Stack Encodings

draft-rosen-tag-stack-02.txt

Status of this Memo

Abstract

"Multi-Protocol Label Switching (MPLS)" [1,2] requires a set of procedures for augmenting network layer packets with "label stacks" (sometimes called "tag stacks"), thereby turning them into "labeled packets". Routers which support MPLS are known as "Label Switching Routers", or "LSRs". In order to transmit a labeled packet on a particular data link, an LSR must support an encoding technique which, given a label stack and a network layer packet, produces a labeled packet. This document specifies the encoding to be used by an LSR in order to transmit labeled packets on PPP data links and on LAN data links. This document also specifies rules and procedures

for processing the various fields of the label stack encoding.

Table of Contents

1. Introduction

"Multi-Protocol Label Switching (MPLS)" [1,2] requires a set of procedures for augmenting network layer packets with "label stacks" (sometimes called "tag stacks"), thereby turning them into "labeled packets". Routers which support MPLS are known as "Label Switching

Routers", or "LSRs". In order to transmit a labeled packet on a particular data link, an LSR must support an encoding technique which, given a label stack and a network layer packet, produces a labeled packet.

This document specifies the encoding to be used by an LSR in order to transmit labeled packets on PPP data links and on LAN data links.

This document also specifies rules and procedures for processing the various fields of the label stack encoding. Since MPLS is independent of any particular network layer protocol, the majority of such procedures are also protocol-independent. A few, however, do differ for different protocols. In this document, we specify the protocol-independent procedures, and we specify the protocol-dependent procedures for IPv4.

LSRs that are implemented on certain switching devices (such as ATM switches) may use different encoding techniques for encoding the top one or two entries of the label stack. When the label stack has additional entries, however, the encoding technique described in this document may be used for the additional label stack entries.

1.1. Specification of Requirements

In this document, several words are used to signify the requirements of the specification. These words are often capitalized.

MUST

This word, or the adjective "required", means that the definition is an absolute requirement of the specification.

MUST NOT

This phrase means that the definition is an absolute prohibition of the specification.

SHOULD

This word, or the adjective "recommended", means that there may

exist valid reasons in particular circumstances to ignore this item, but the full implications must be understood and carefully weighed before choosing a different course.

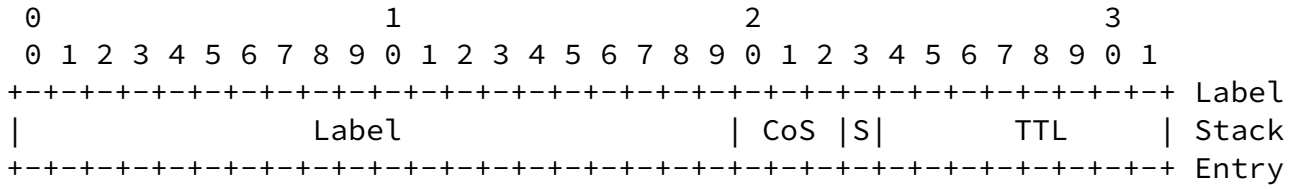
MAY

This word, or the adjective "optional", means that this item is one of an allowed set of alternatives. An implementation which does not include this option MUST be prepared to interoperate with another implementation which does include the option.

2. The Label Stack

2.1. Encoding the Label Stack

On both PPP and LAN data links, the label stack is represented as a sequence of "label stack entries". Each label stack entry is represented by 4 octets. This is shown in Figure 1.



Label: Label Value, 20 bits
CoS: Class of Service, 3 bits
S: Bottom of Stack, 1 bit
TTL: Time to Live, 8 bits

Figure 1

The label stack entries appear AFTER the data link layer headers, but BEFORE any network layer headers. The top of the label stack appears earliest in the packet, and the bottom appears latest. The network layer packet immediately follows the label stack entry which has the S bit set.

Each label stack entry is broken down into the following fields:

1. Bottom of Stack (S)

   This bit is set to one for the last entry in the label stack (i.e., for the bottom of the stack), and zero for all other label stack entries.

2. Time to Live (TTL)

   This eight-bit field is used to encode a time-to-live value. The processing of this field is described in section 2.3.

3. Class of Service (CoS)

   This three-bit field is used to identify a "Class of Service". The setting of this field is intended to affect the scheduling and/or discard algorithms which are applied to the packet as it is transmitted through the network.

When an unlabeled packet is initially labeled, the value assigned to the CoS field in the label stack entry is determined by policy. Some possible policies are:

- the CoS value is a function of the IP ToS value
- the CoS value is a function of the packet's input interface
- the CoS value is a function of the "flow type"

Of course, many other policies are also possible.

When an additional label is pushed onto the stack of a packet that is already labeled:

- in general, the value of the CoS field in the new top stack entry should be equal to the value of the CoS field of the old top stack entry;
- however, in some cases, most likely at boundaries between network service providers, the value of the CoS field in the new top stack entry may be determined by policy.

4. Label Value

This 20-bit field carries the actual value of the Label.

When a labeled packet is received, the label value at the top of the stack is looked up. As a result of this lookup one learns:

(a) information needed to forward the packet, such as the next hop and the outgoing data link encapsulation; however, the precise queue to put the packet on, or information as to how to schedule the packet, may be a function of both the label value AND the CoS field value;

(b) the operation to be performed on the label stack before forwarding; this operation may be to replace the top label stack entry with another, or to pop an entry off the label stack, or to replace the top label stack entry and then to push one or more additional entries on the label stack.

There are several reserved label values:

i. A value of 0 represents the "IPv4 Explicit NULL Label". This label value is only legal when it is the sole label stack entry. It indicates that the label stack must be popped, and the forwarding of the packet must then be based on the IPv4 header.

ii. A value of 1 represents the "Router Alert Label". This label value is legal anywhere in the label stack except at the bottom. When a received packet contains this label value at the top of the label stack, it is delivered to a local software module for processing. The actual forwarding of the packet is determined by the label beneath it in the stack. However, if the packet is forwarded further, the Router Alert Label should be pushed back onto the label stack before

forwarding. The use of this label is analogous to the use of the "Router Alert Option" in IP packets [6]. Since this label cannot occur at the bottom of the stack, it is not associated with a particular network layer protocol.

iii. A value of 2 represents the "IPv6 Explicit NULL Label". This label value is only legal when it is the sole label stack entry. It indicates that the label stack must be popped, and the forwarding of the packet must then be based on the IPv6 header.

iv. Values 3-16 are reserved.

We must also discuss the "Implicit NULL Label". This is a label that an LSR may assign and distribute, but which never actually appears in the encapsulation. When an LSR would otherwise replace the label at the top of the stack with a new label, but the new label is "Implicit NULL", the LSR will pop the stack instead of doing the replacement.

2.2. Determining the Network Layer Protocol

When the last label is popped from the label stack, it is necessary to determine the particular network layer protocol which is being carried. Note that the label stack entries carry no explicit field to identify the network layer header. Rather, this must be inferable from the value of the label which is popped from the bottom of the stack. This means that when the first label is pushed onto a network layer packet, the label must be one which is used ONLY for packets of a particular network layer. Furthermore, whenever that label is replaced by another label value during a packet's transit, the new

value must also be one which is used only for packets of that network layer.

2.3. Processing the Time to Live Field

2.3.1. Definitions

The "incoming TTL" of a labeled packet is defined to be the value of the TTL field of the top label stack entry when the packet is received.

The "outgoing TTL" of a labeled packet is defined to be the larger of:

(a) one less than the incoming TTL,
(b) zero.

2.3.2. Protocol-independent rules

If the outgoing TTL of a labeled packet is 0, then the labeled packet MUST NOT be further forwarded; the packet's lifetime in the network is considered to have expired.

Depending on the label value in the label stack entry, the packet MAY be silently discarded, or the packet MAY have its label stack stripped off, and passed as an unlabeled packet to the ordinary processing for network layer packets which have exceeded their maximum lifetime in the network. However, even if the label stack is stripped, the packet MUST NOT be further forwarded.

When a labeled packet is forwarded, the TTL field of the label stack entry at the top of the label stack must be set to the outgoing TTL value.

Note that the outgoing TTL value is a function solely of the incoming TTL value, and is independent of whether any labels are pushed or popped before forwarding. There is no significance to the value of the TTL field in any label stack entry which is not at the top of the stack.

2.3.3. IP-dependent rules

When an IP packet is first labeled, the TTL field of the label stack entry MUST BE set to the value of the IP TTL field. (If the IP TTL field needs to be decremented, as part of the IP processing, it is assumed that this has already been done.)

When a label is popped, and the resulting label stack is empty, then the value of the IP TTL field MUST BE replaced with the outgoing TTL value, as defined above. Note that, in IPv4, this will also require modification of the IP header checksum.

3. Fragmentation and Path MTU Discovery

Just as it is possible to receive an unlabeled IP datagram which is too large to be transmitted on its output link, it is possible to receive a labeled packet which is too large to be transmitted on its output link.

It is also possible that a received packet (labeled or unlabeled) which was originally small enough to be transmitted on that link becomes too large by virtue of having one or more additional labels pushed onto its label stack. In label switching, a packet may grow in size if additional labels get pushed on. Thus if one receives a labeled packet with a 1500-byte frame payload, and pushes on an additional label, one needs to forward it as frame with a 1504-byte payload.

This section specifies the rules for processing labeled packets which are "too large". In particular, it provides rules which ensure that hosts implementing RFC 1191 Path MTU Discovery will be able to generate IP datagrams that do not need fragmentation, even if they get labeled as the traverse the network.

In general, hosts which do not implement RFC 1191 Path MTU Discovery send IP datagrams which contain no more than 576 bytes. Since the MTUs in use on most data links today are 1500 bytes or more, the probability that such datagrams will need to get fragmented, even if they get labeled, is very small.

Some hosts that do not implement RFC 1191 Path MTU Discovery will generate IP datagrams containing 1500 bytes, as long as the IP Source and Destination addresses are on the same subnet. These datagrams will not pass through routers, and hence will not get fragmented.

Unfortunately, some hosts will generate IP datagrams containing 1500 bytes, as long the IP Source and Destination addresses do not have

the same classful network number. This is the one case in which there is significant risk of fragmentation when such datagrams get labeled.

This document specifies procedures which allow one to configure the network so that large datagrams from hosts which do not implement Path MTU Discovery get fragmented just once, when they are first labeled. These procedures make it possible (assuming suitable configuration) to avoid any need to fragment packets which have already been labeled.

3.1. Terminology

With respect to a particular data link, we can use the following terms:

- Frame Payload:

  The contents of a data link frame, excluding any data link layer headers or trailers (e.g., MAC headers, LLC headers, 802.1Q or 802.1p headers, PPP header, frame check sequences, etc.).

  When a frame is carrying an an unlabeled IP datagram, the Frame Payload is just the IP datagram itself. When a frame is carrying a labeled IP datagram, the Frame Payload consists of the label stack entries and the IP datagram.

- Conventional Maximum Frame Payload Size:

  The maximum Frame Payload size allowed by data link standards. For example, the Conventional Maximum Frame Payload Size for ethernet is 1500 bytes.

- True Maximum Frame Payload Size:

  The maximum size frame payload which can be sent and received properly by the interface hardware attached to the data link.

  On ethernet and 802.3 networks, it is believed that the True Maximum Frame Payload Size is 4-8 bytes larger than the Conventional Maximum Frame Payload Size (as long neither an 802.1Q header nor an 802.1p header is present, and as long as neither can be added by a switch or bridge while a packet is in transit to its next hop). For example, it is believed that most ethernet equipment could correctly send and receive packets

carrying a payload of 1504 or perhaps even 1508 bytes, at least, as long as the ethernet header does not have an 802.1Q or 802.1p

field.

On PPP links, the True Maximum Frame Payload Size may be virtually unbounded.

- Effective Maximum Frame Payload Size for Labeled Packets:

  This is either be the Conventional Maximum Frame Payload Size or the True Maximum Frame Payload Size, depending on the capabilities of the equipment on the data link and the size of the ethernet header being used.

- Initially Labeled IP Datagram

  Suppose that an unlabeled IP datagram is received at a particular LSR, and that the the LSR pushes on a label before forwarding the datagram. Such a datagram will be called an Initially Labeled IP Datagram at that LSR.

- Previously Labeled IP Datagram

  An IP datagram which had already been labeled before it was received by a particular LSR.

3.2. Maximum Initially Labeled IP Datagram Size

Every LSR which is capable of

(a) receiving an unlabeled IP datagram,
(b) adding a label stack to the datagram, and
(c) forwarding the resulting labeled packet,

MUST support a configuration parameter known as the "Maximum IP Datagram Size for Labeling", which can be set to a non-negative value.

If this configuration parameter is set to zero, it has no effect.

If it is set to a positive value, it is used in the following way. If:

(a) an unlabeled IP datagram is received, and
(b) that datagram does not have the DF bit set in its IP header, and
(c) that datagram needs to be labeled before being forwarded, and

(d) the size of the datagram (before labeling) exceeds the value of the parameter,

then

(a) the datagram must be broken into fragments, each of whose size is no greater than the value of the parameter, and
(b) each fragment must be labeled and then forwarded.

If this configuration parameter is set to a value of 1488, for example, then any unlabeled IP datagram containing more than 1488 bytes will be fragmented before being labeled. Each fragment will be capable of being carried on a 1500-byte data link, without further fragmentation, even if as many as three labels are pushed onto its label stack.

In other words, setting this parameter to a non-zero value allows one to eliminate all fragmentation of Previously Labeled IP Datagrams, but it may cause some unnecessary fragmentation of Initially Labeled IP Datagrams.

Note that the parameter has no effect on IP Datagrams that have the DF bit set, which means that it has no effect on Path MTU Discovery.

3.3. When are Labeled IP Datagrams Too Big?

A labeled IP datagram whose size exceeds the Conventional Maximum Frame Payload Size of the data link over which it is to be forwarded MAY be considered to be "too big".

A labeled IP datagram whose size exceeds the True Maximum Frame Payload Size of the data link over which it is to be forwarded MUST be considered to be "too big".

A labeled IP datagram which is not "too big" MUST be transmitted without fragmentation.

3.4. Processing Labeled IP Datagrams which are Too Big

If a labeled IP datagram is "too big", and the DF bit is not set in its IP header, then the LSR MAY discard the datagram.

Note that discarding such datagrams is a sensible procedure only if the "Maximum Initially Labeled IP Datagram Size" is set to a non-zero value in every LSR in the network which is capable of adding a label stack to an unlabeled IP datagram.

If the LSR chooses not to discard a labeled IP datagram which is too

big, or if the DF bit is set in that datagram, then it MUST execute the following algorithm:

1. Strip off the label stack entries to obtain the IP datagram.
2. Let N be the number of bytes in the label stack (i.e, 4 times the number of label stack entries).
3. If the IP datagram does NOT have the "Don't Fragment" bit set in its IP header:
    a. convert it into fragments, each of which MUST be at least N bytes less than the Effective Maximum Frame Payload Size.
    b. Prepend each fragment with the same label header that would have been on the original datagram had fragmentation not been necessary.
    c. Forward the fragments
4. If the IP datagram has the "Don't Fragment" bit set in its IP header:
    a. the datagram MUST NOT be forwarded

b. Create an ICMP Destination Unreachable Message:

i. set its Code field (RFC 792) to "Fragmentation Required and DF Set",

ii. set its Next-Hop MTU field (RFC 1191) to the difference between the Effective Maximum Frame Payload Size and the value of N

c. If possible, transmit the ICMP Destination Unreachable Message to the source of the of the discarded datagram.

3.5. Implications with respect to Path MTU Discovery

The procedures described above for handling datagrams which have the DF bit set, but which are "too large", have an impact on the Path MTU Discovery procedures of RFC 1191. Hosts which implement these procedures will discover an MTU which is small enough to allow n labels to be pushed on the datagrams, without need for fragmentation, where n is the number of labels that actually get pushed on along the path currently in use.

In other words, datagrams from hosts that use Path MTU Discovery will never need to be fragmented due to the need to put on a label header, or to add new labels to an existing label header. (Also, datagrams from hosts that use Path MTU Discovery generally have the DF bit set, and so will never get fragmented anyway.)

However, note that Path MTU Discovery will only work properly if, at the point where a labeled IP Datagram's fragmentation needs to occur, it is possible to route to the packet's source address. If this is not possible, then the ICMP Destination Unreachable message cannot be sent to the source.

3.5.1. Tunneling through a Transit Routing Domain

Suppose one is using MPLS to "tunnel" through a transit routing domain, where the external routes are not leaked into the domain's interior routers. If a packet needs fragmentation at some router

within the domain, and the packet's DF bit is set, it is necessary to be able to originate an ICMP message at that router and have it routed correctly to the source of the fragmented packet. If the packet's source address is an external address, this poses a problem.

Therefore, in order for Path MTU Discovery to work, any routing domain in which external routes are not leaked into the interior routers MUST have a default route which causes all packets carrying external destination addresses to be sent to a border router. For example, one of the border routers may inject "default" into the IGP.

3.5.2. Tunneling Private Addresses through a Public Backbone

In other cases where MPLS is used to tunnel through a routing domain, it may not be possible to route to the source address of a fragmented packet at all. This would be the case, for example, if the IP addresses carried in the packet were private addresses, and MPLS were being used to tunnel those packets through a public backbone.

In such cases, the LSR at the transmitting end of the tunnel MUST be able to determine the MTU of the tunnel as a whole. It SHOULD do this by sending packets through the tunnel to the tunnel's receiving endpoint, and performing Path MTU Discovery with those packets. Then any time the transmitting endpoint of the tunnel needs to send a packet into the tunnel, and that packet has the DF bit set, and it exceeds the tunnel MTU, the transmitting endpoint of the tunnel MUST send the ICMP Destination Unreachable message to the source, with code "Fragmentation Required and DF Set", and the Next-Hop MTU Field set as described above.

4. Transporting Labeled Packets over PPP

The Point-to-Point Protocol (PPP) [PPP] provides a standard method for transporting multi-protocol datagrams over point-to-point links. PPP defines an extensible Link Control Protocol, and proposes a family of Network Control Protocols for establishing and configuring different network-layer protocols.

This section defines the Network Control Protocol for establishing and configuring label Switching over PPP.

4.1. Introduction

PPP has three main components:

1. A method for encapsulating multi-protocol datagrams.

2. A Link Control Protocol (LCP) for establishing, configuring, and testing the data-link connection.

3. A family of Network Control Protocols for establishing and configuring different network-layer protocols.

In order to establish communications over a point-to-point link, each end of the PPP link must first send LCP packets to configure and test the data link. After the link has been established and optional facilities have been negotiated as needed by the LCP, PPP must send "MPLS Control Protocol" packets to enable the transmission of labeled packets. Once the "MPLS Control Protocol" has reached the Opened state, labeled packets can be sent over the link.

The link will remain configured for communications until explicit LCP or MPLS Control Protocol packets close the link down, or until some external event occurs (an inactivity timer expires or network administrator intervention).

4.2. A PPP Network Control Protocol for MPLS

The MPLS Control Protocol (MPLSCP) is responsible for enabling and disabling the use of label switching on a PPP link. It uses the same packet exchange mechanism as the Link Control Protocol (LCP). MPLSCP packets may not be exchanged until PPP has reached the Network-Layer Protocol phase. MPLSCP packets received before this phase is reached should be silently discarded.

The MPLS Control Protocol is exactly the same as the Link Control

Protocol [7] with the following exceptions:

1. Frame Modifications

The packet may utilize any modifications to the basic frame format which have been negotiated during the Link Establishment phase.

2. Data Link Layer Protocol Field

   Exactly one MPLSCP packet is encapsulated in the PPP Information field, where the PPP Protocol field indicates type hex 8081 (MPLS).

3. Code field

   Only Codes 1 through 7 (Configure-Request, Configure-Ack, Configure-Nak, Configure-Reject, Terminate-Request, Terminate-Ack and Code-Reject) are used. Other Codes should be treated as unrecognized and should result in Code-Rejects.

4. Timeouts

   MPLSCP packets may not be exchanged until PPP has reached the Network-Layer Protocol phase. An implementation should be prepared to wait for Authentication and Link Quality Determination to finish before timing out waiting for a Configure-Ack or other response. It is suggested that an implementation give up only after user intervention or a configurable amount of time.

5. Configuration Option Types

   None.

4.3. Sending Labeled Packets

Before any labeled packets may be communicated, PPP must reach the Network-Layer Protocol phase, and the MPLS Control Protocol must reach the Opened state.

Exactly one labeled packet is encapsulated in the PPP Information field, where the PPP Protocol field indicates either type hex 0081 (MPLS Unicast) or type hex 0083 (MPLS Multicast). The maximum length of a labeled packet transmitted over a PPP link is the same as the maximum length of the Information field of a PPP encapsulated packet.

The format of the Information field itself is as defined in section 2.

Note that two codepoints are defined for labeled packets; one for multicast and one for unicast. Once the MPLSCP has reached the Opened state, both label Switched multicasts and label Switched unicasts can be sent over the PPP link.

4.4. Label Switching Control Protocol Configuration Options

There are no configuration options.

5. Transporting Labeled Packets over LAN Media

Exactly one labeled packet is carried in each frame.

The label stack entries immediately precede the network layer header, and follow any data link layer headers, including any VLAN headers, 802.1p headers, and/or 802.1Q headers that may exist.

The ethertype value 8847 hex is used to indicate that a frame is carrying an MPLS unicast packet.

The ethertype value 8848 hex is used to indicate that a frame is carrying an MPLS multicast packet.

These ethertype values can be used with either the ethernet encapsulation or the 802.3 SNAP/SAP encapsulation to carry labeled packets.

6. Security Considerations

Security considerations are not discussed in this document.

7. Authors' Addresses

Eric C. Rosen
Cisco Systems, Inc.
250 Apollo Drive
Chelmsford, MA, 01824

E-mail: erosen@cisco.com

Dan Tappan
Cisco Systems, Inc.
250 Apollo Drive
Chelmsford, MA, 01824

E-mail: tappan@cisco.com

Dino Farinacci
Cisco Systems, Inc.
170 Tasman Drive
San Jose, CA, 95134

E-mail: dino@cisco.com

Yakov Rekhter
Cisco Systems, Inc.
170 Tasman Drive
San Jose, CA, 95134

E-mail: yakov@cisco.com

Guy Fedorkow
Cisco Systems, Inc.
250 Apollo Drive
Chelmsford, MA, 01824

E-mail: fedorkow@cisco.com

Tony Li
Juniper Networks
3260 Jay Street
Santa Clara, CA 95051

E-mail: tli@jnx.com

8. References

[1] "Tag Switching Architecture - Overview", 1/9/97, draft-rekhter-tagswitch-arch-00.txt, Rekhter, Davie, Katz, Rosen, Swallow

[2] "A Framework for Multiprotocol Label Switching", 5/12/97, draft-ietf-mpls-framework-00.txt, Callon, Doolan, Feldman, Fredette, Swallow, Visanathawan

[3] "Internet Protocol", RFC 791, 9/81, Postel

[4] "Internet Control Message Protocol", RFC 792, 9/81, Postel

[5] "Path MTU Discovery", RFC 1191, 11/90, Mogul & Deering

[6] "IP Router Alert Option", RFC 2113, 2/97, Katz

[7] "The Point-to-Point Protocol (PPP)", RFC 1661, 7/94, Simpson