Network Working Group   Kireeti Kompella
Internet Draft   Juniper Networks, Inc.
Category: Standards Track
Expiration Date: November 2005

                                                   George Swallow
                                              Cisco Systems, Inc.

                                                         May 2005

Detecting MPLS Data Plane Failures

draft-ietf-mpls-lsp-ping-09.txt

Status of this Memo

Abstract

   This document describes a simple and efficient mechanism that can be
   used to detect data plane failures in Multi-Protocol Label Switching
   (MPLS) Label Switched Paths (LSPs). There are two parts to this
   document: information carried in an MPLS "echo request" and "echo
   reply" for the purposes of fault detection and isolation; and
   mechanisms for reliably sending the echo reply.

Changes since last revision

   (This section to be removed before publication.)

    o fixed the optional vs. mandatory TLV wording
    o Removed the Error TLV (which never found use and was undefined)
    o Removed an extraneous paragraph from the processing rules and did
      some further word-smithing
    o Removed Appendix A on CR-LDP
    o Added an Address Type field to the Address and Interface Object;
      combined the IPv4 & IPv6 formats into a single TLV
    o Modified example TLV / Sub-TLV to be more instructive on
      determining (Sub-)TLV lengths
    o Removed the BGP Next Hop field from the BGP Labeled IPv4 FEC
    o Added the BGP Labeled IPv6 FEC
    o Corrected the length calculation for the Downsteam Mapping TLV
    o Added two special values for the Downstream Router ID
    o Added an error code for upstream Interface Index Unknown
    o Added all requested allocations to the IANA section
    o Spelling and Word-smithing

Contents

1. Introduction

This document describes a simple and efficient mechanism that can be
used to detect data plane failures in MPLS LSPs. There are two parts
to this document: information carried in an MPLS "echo request" and
"echo reply"; and mechanisms for transporting the echo reply. The
first part aims at providing enough information to check correct
operation of the data plane, as well as a mechanism to verify the
data plane against the control plane, and thereby localize faults.
The second part suggests two methods of reliable reply channels for
the echo request message, for more robust fault isolation.

An important consideration in this design is that MPLS echo requests
follow the same data path that normal MPLS packets would traverse.
MPLS echo requests are meant primarily to validate the data plane,
and secondarily to verify the data plane against the control plane.
Mechanisms to check the control plane are valuable, but are not cov-
ered in this document.

1.1. Conventions

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 RFC 2119 [KEYWORDS].

1.2. Structure of this document

The body of this memo contains four main parts: motivation, MPLS echo
request/reply packet format, LSP ping operation, and a reliable
return path. It is suggested that first-time readers skip the actual
packet formats and read the Theory of Operation first; the document
is structured the way it is to avoid forward references.

1.3. Contributors

The following made vital contributions to all aspects of this docu-
ment, and much of the material came out of debate and discussion
among this group.

    Ronald P. Bonica, Juniper Networks, Inc.
    Dave Cooper, Global Crossing
    Ping Pan, Hammerhead Systems
    Nischal Sheth, Juniper Networks, Inc.
    Sanjay Wadhwa, Juniper Networks, Inc.

2. Motivation

   When an LSP fails to deliver user traffic, the failure cannot always
   be detected by the MPLS control plane. There is a need to provide a
   tool that would enable users to detect such traffic "black holes" or
   misrouting within a reasonable period of time; and a mechanism to
   isolate faults.

   In this document, we describe a mechanism that accomplishes these
   goals. This mechanism is modeled after the ping/traceroute paradigm:
   ping (ICMP echo request [ICMP]) is used for connectivity checks, and
   traceroute is used for hop-by-hop fault localization as well as path
   tracing. This document specifies a "ping mode" and a "traceroute"
   mode for testing MPLS LSPs.

   The basic idea is to verify that packets that belong to a particular
   Forwarding Equivalence Class (FEC) actually end their MPLS path on an
   LSR that is an egress for that FEC. This document proposes that this
   test be carried out by sending a packet (called an "MPLS echo
   request") along the same data path as other packets belonging to this
   FEC. An MPLS echo request also carries information about the FEC
   whose MPLS path is being verified. This echo request is forwarded
   just like any other packet belonging to that FEC. In "ping" mode
   (basic connectivity check), the packet should reach the end of the
   path, at which point it is sent to the control plane of the egress
   LSR, which then verifies whether it is indeed an egress for the FEC.
   In "traceroute" mode (fault isolation), the packet is sent to the
   control plane of each transit LSR, which performs various checks that
   it is indeed a transit LSR for this path; this LSR also returns fur-
   ther information that helps check the control plane against the data
   plane, i.e., that forwarding matches what the routing protocols
   determined as the path.

   One way these tools can be used is to periodically ping a FEC to
   ensure connectivity. If the ping fails, one can then initiate a
   traceroute to determine where the fault lies. One can also periodi-
   cally traceroute FECs to verify that forwarding matches the control
   plane; however, this places a greater burden on transit LSRs and thus
   should be used with caution.

[3](3). Packet Format

   An MPLS echo request is a (possibly labeled) IPv4 or IPv6 UDP packet;
   the contents of the UDP packet have the following format:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |         Version Number        |         Global Flags          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  Message Type |   Reply mode  |  Return Code  | Return Subcode|
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                         Sender's Handle                       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                         Sequence Number                       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                     TimeStamp Sent (seconds)                  |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                  TimeStamp Sent (microseconds)                |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                   TimeStamp Received (seconds)                |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                 TimeStamp Received (microseconds)             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                           TLVs ...                            |
   .                                                              .
   .                                                              .
   .                                                              .
   |                                                              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
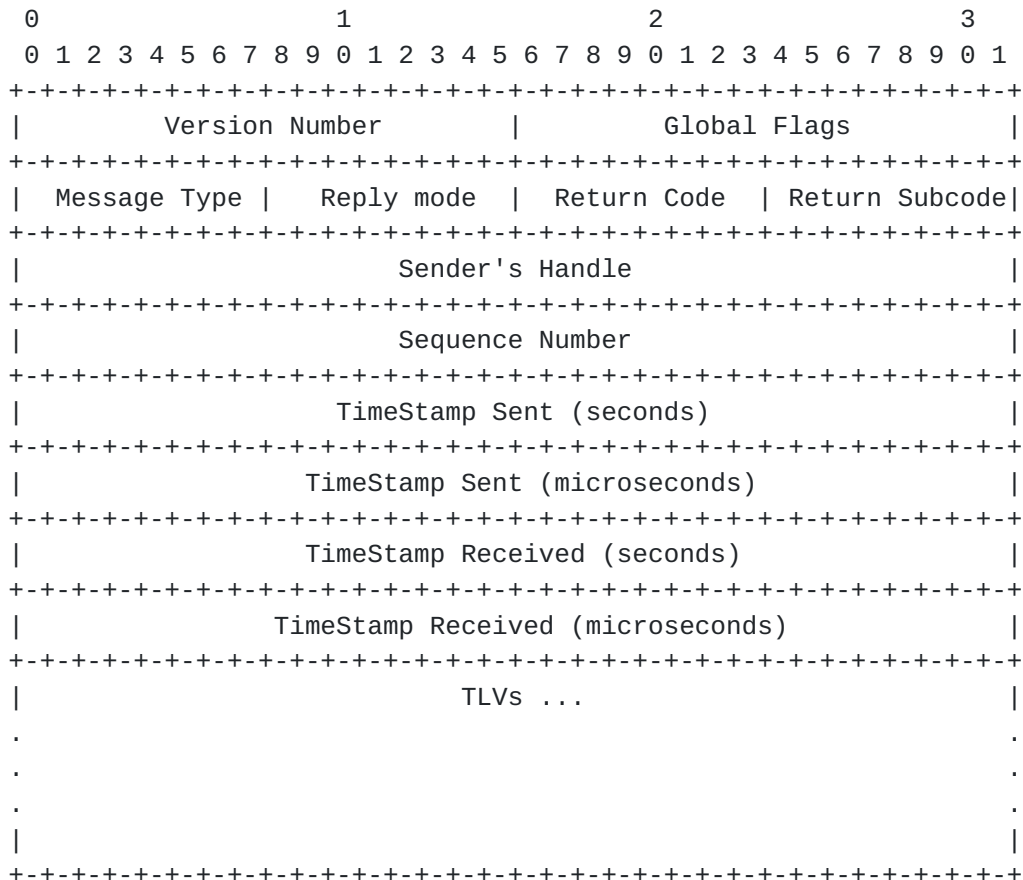
   The Version Number is currently 1. (Note: the Version Number is to
   be incremented whenever a change is made that affects the ability of
   an implementation to correctly parse or process an MPLS echo
   request/reply. These changes include any syntactic or semantic
   changes made to any of the fixed fields, or to any TLV or sub-TLV
   assignment or format that is defined at a certain version number.
   The Version Number may not need to be changed if an optional TLV or
   sub-TLV is added.)

   The Global Flags field is a bit vector with the following format:

```
     0                   1
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |            MBZ            |V|
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
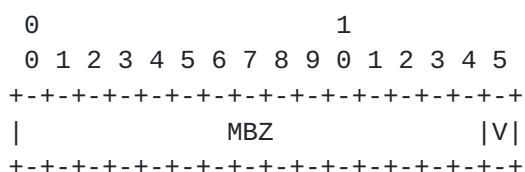
One flag is defined for now, the V bit; the rest MUST be set to zero
when sending, and ignored on receipt.

The V (Validate FEC Stack) flag is set to 1 if the sender wants the
receiver to perform FEC stack validation; if V is 0, the choice is
left to the receiver.

The Message Type is one of the following:

```
    Value    Meaning
    -----    -------
        1    MPLS Echo Request
        2    MPLS Echo Reply
```

The Reply Mode can take one of the following values:

```
    Value    Meaning
    -----    -------
        1    Do not reply
        2    Reply via an IPv4/IPv6 UDP packet
        3    Reply via an IPv4/IPv6 UDP packet with Router Alert
        4    Reply via application level control channel
```

An MPLS echo request with "Do not reply" may be used for one-way con-
nectivity tests; the receiving router may log gaps in the sequence
numbers and/or maintain delay/jitter statistics. An MPLS echo
request would normally have "Reply via an IPv4/IPv6 UDP packet"; if
the normal IP return path is deemed unreliable, one may use "Reply
via an IPv4/IPv6 UDP packet with Router Alert" (note that this
requires that all intermediate routers understand and know how to
forward MPLS echo replies). The echo reply uses the same IP version
number as the received echo request, i.e., an IPv4 encapsulated echo
reply is sent in response to an IPv4 encapsulated echo request.

Any application which supports an IP control channel between its con-
trol entities may set the Reply Mode to 4 to ensure that replies use
that same channel. Further definition of this codepoint is applica-
tion specific and thus beyond the scope of this document.

Return Codes and Subcodes are described in the next section.

the Sender's Handle is filled in by the sender, and returned
unchanged by the receiver in the echo reply (if any). There are no
semantics associated with this handle, although a sender may find
this useful for matching up requests with replies.

The Sequence Number is assigned by the sender of the MPLS echo
request, and can be (for example) used to detect missed replies.

The TimeStamp Sent is the time-of-day (in seconds and microseconds,
wrt the sender's clock) when the MPLS echo request is sent. The
TimeStamp Received in an echo reply is the time-of-day (wrt the
receiver's clock) that the corresponding echo request was received.

TLVs (Type-Length-Value tuples) have the following format:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |             Type              |            Length             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                             Value                             |
   .                                                               .
   .                                                               .
   .                                                               .
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Types are defined below; Length is the length of the Value field in
octets. The Value field depends on the Type; it is zero padded to
align to a four-octet boundary. TLVs may be nested within other
TLVs, in which case the nested TLVs are called sub-TLVs. Sub-TLVs
have independent types and MUST also be four-octet aligned.

Two examples follow. The LDP IPv4 FEC sub-TLV has the following for-
mat:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |    Type = 1 (LDP IPv4 FEC)    |          Length = 5           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                          IPv4 prefix                          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Prefix Length |          Must Be Zero                         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Length for this TLV is 5. A Target FEC Stack TLV which contains
an LDP IPv4 FEC sub-TLV and a VPN IPv4 FEC sub-TLV has the format:

```
      0                   1                   2                   3
       0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |       Type = 1 (FEC TLV)       |           Length = 12         |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |   sub-Type = 1 (LDP IPv4 FEC)  |          Length = 5           |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                          IPv4 prefix                          |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      | Prefix Length |          Must Be Zero                         |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |  sub-Type = 6 (VPN IPv4 FEC)   |          Length = 13          |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                       Route Distinguisher                     |
      |                           (8 octets)                          |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                          IPv4 prefix                          |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      | Prefix Length |              Must Be Zero                     |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

A description of the Types and Values of the top level TLVs for LSP
ping are given below:

```
           Type #                  Value Field
           ------                  -----------
                1                  Target FEC Stack
                2                  Downstream Mapping
                3                  Pad
                4                  Not Assigned
                5                  Vendor Enterprise Code
                6                  Not Assigned
                7                  Interface and Label Stack
                8                  Not Assigned
                9                  Errored TLVs
               10                  Reply TOS Byte
```

Types less than 32768 (i.e., with the high order bit equal to 0) are
mandatory TLVs that MUST either be supported by an implementation or
result in the return code of 2 ("One or more of the TLVs was not
understood") being sent in the echo response.

Types greater than or equal to 32768 (i.e., with the high order bit
equal to 1) are optional TLVs that SHOULD be ignored if the implemen-
tation does not understand or support them.

[3.1](3.1). **Return Codes**

   The Return Code is set to zero by the sender. The receiver can set
   it to one of the values listed below. The notation <RSC> refers to
   the Return Subcode. This field is filled in with the stack-depth for
   those codes which specify that. For all other codes the Return Sub-
   code MUST be set to zero.

        Value   Meaning
        -----   -------

          0   No return code

          1   Malformed echo request received

          2   One or more of the TLVs was not understood

          3   Replying router is an egress for the FEC at stack
               depth <RSC>

          4   Replying router has no mapping for the FEC at stack
               depth <RSC>

          5   Downstream Mapping Mismatch (See Note 1)

          6   Upstream Interface Index Unknown (See Note 1)

          7   Reserved

          8   Label switched at stack-depth <RSC>

          9   Label switched but no MPLS forwarding at stack-depth
               <RSC>

         10   Mapping for this FEC is not the given label at stack
               depth <RSC>

         11   No label entry at stack-depth <RSC>

         12   Protocol not associated with interface at FEC stack
               depth <RSC>

         13   Premature termination of ping due to label stack
               shrinking to a single label

Note 1

   The Return Subcode contains the point in the label stack where pro-
   cessing was terminated. If the RSC is 0, no labels were processed.
   Otherwise the packet would have been label switched at depth RSC.

3.2. **Target FEC Stack**

   A Target FEC Stack is a list of sub-TLVs. The number of elements is
   determined by the looking at the sub-TLV length fields.

Sub-Type	Length	Value Field
1	5	LDP IPv4 prefix
2	17	LDP IPv6 prefix
3	20	RSVP IPv4 LSP
4	56	RSVP IPv6 LSP
5		Not Assigned
6	13	VPN IPv4 prefix
7	25	VPN IPv6 prefix
8	14	L2 VPN endpoint
9	10	"FEC 128" Pseudowire (deprecated)
10	14	"FEC 128" Pseudowire
11	13+	"FEC 129" Pseudowire
12	5	BGP labeled IPv4 prefix
13	17	BGP labeled IPv6 prefix
14	5	Generic IPv4 prefix
15	17	Generic IPv6 prefix
16	4*N	Nil FEC

   Other FEC Types will be defined as needed.

   Note that this TLV defines a stack of FECs, the first FEC element
   corresponding to the top of the label stack, etc.

   An MPLS echo request MUST have a Target FEC Stack that describes the
   FEC stack being tested. For example, if an LSR X has an LDP mapping
   for 192.168.1.1 (say label 1001), then to verify that label 1001 does
   indeed reach an egress LSR that announced this prefix via LDP, X can
   send an MPLS echo request with a FEC Stack TLV with one FEC in it,
   namely of type LDP IPv4 prefix, with prefix 192.168.1.1/32, and send
   the echo request with a label of 1001.

   Say LSR X wanted to verify that a label stack of <1001, 23456> is the
   right label stack to use to reach a VPN IPv4 prefix of 10/8 in VPN
   foo. Say further that LSR Y with loopback address 192.168.1.1
   announced prefix 10/8 with Route Distinguisher RD-foo-Y (which may in

general be different from the Route Distinguisher that LSR X uses in
its own advertisements for VPN foo), label 23456 and BGP nexthop
192.168.1.1. Finally, suppose that LSR X receives a label binding of
1001 for 192.168.1.1 via LDP. X has two choices in sending an MPLS
echo request: X can send an MPLS echo request with a FEC Stack TLV
with a single FEC of type VPN IPv4 prefix with a prefix of 10/8 and a
Route Distinguisher of RD-foo-Y. Alternatively, X can send a FEC
Stack TLV with two FECs, the first of type LDP IPv4 with a prefix of
192.168.1.1/32 and the second of type of IP VPN with a prefix 10/8
with Route Distinguisher of RD-foo-Y. In either case, the MPLS echo
request would have a label stack of <1001, 23456>. (Note: in this
example, 1001 is the "outer" label and 23456 is the "inner" label.)

3.2.1. LDP IPv4 Prefix

The value consists of four octets of an IPv4 prefix followed by one
octet of prefix length in bits; the format is given below. The IPv4
prefix is in network byte order; if the prefix is shorter than 32
bits, trailing bits SHOULD be set to zero. See [LDP] for an example
of a Mapping for an IPv4 FEC.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                          IPv4 prefix                          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Prefix Length |        Must Be Zero                          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
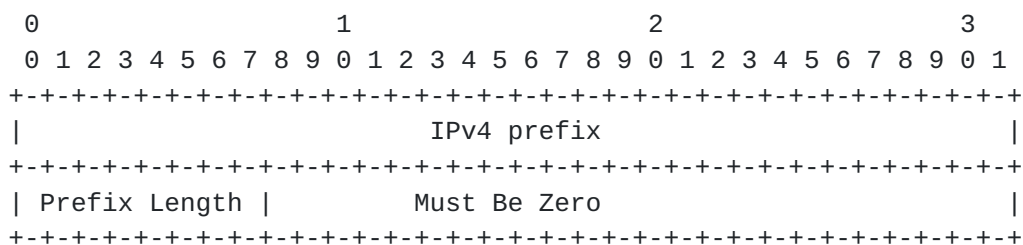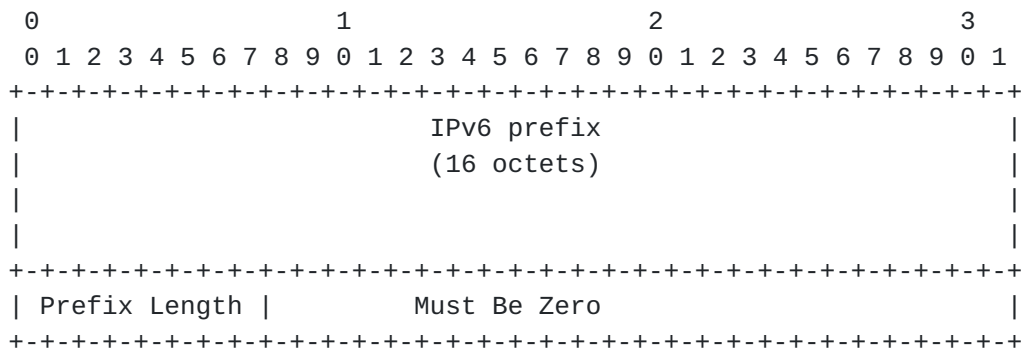
3.2.2. LDP IPv6 Prefix

The value consists of sixteen octets of an IPv6 prefix followed by
one octet of prefix length in bits; the format is given below. The
IPv6 prefix is in network byte order; if the prefix is shorter than
128 bits, the trailing bits SHOULD be set to zero. See [LDP] for an
example of a Mapping for an IPv6 FEC.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         IPv6 prefix                           |
|                         (16 octets)                           |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Prefix Length |           Must Be Zero                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

3.2.3. RSVP IPv4 LSP

The value has the format below. The value fields are taken from
[RFC3209, sections 4.6.1.1 and 4.6.2.1].

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  IPv4 tunnel end point address                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Must Be Zero         |          Tunnel ID            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Extended Tunnel ID                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    IPv4 tunnel sender address                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Must Be Zero         |            LSP ID             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
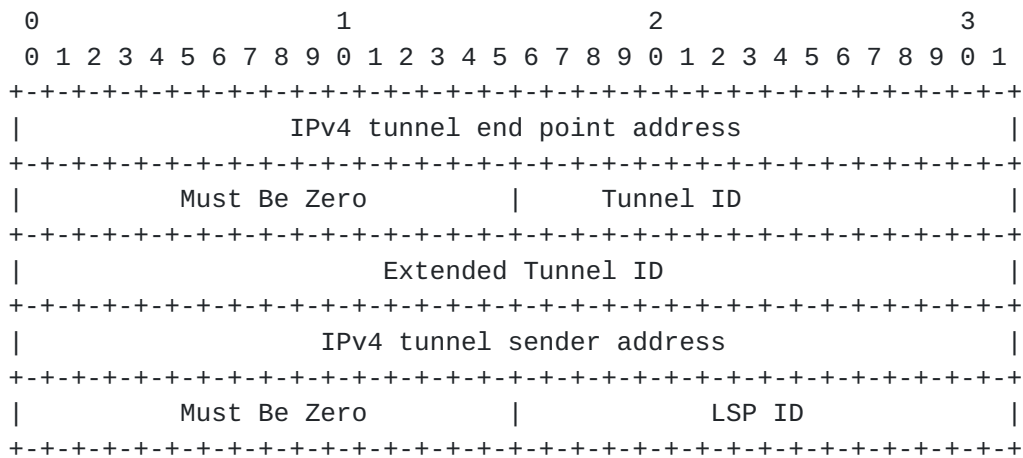
3.2.4. RSVP IPv6 LSP

The value has the format below. The value fields are taken from
[RFC3209, sections 4.6.1.2 and 4.6.2.2].

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                 IPv6 tunnel end point address                 |
   |                                                               |
   |                                                               |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |           Must Be Zero          |           Tunnel ID         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      Extended Tunnel ID                       |
   |                                                               |
   |                                                               |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                   IPv6 tunnel sender address                  |
   |                                                               |
   |                                                               |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |           Must Be Zero          |             LSP ID          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
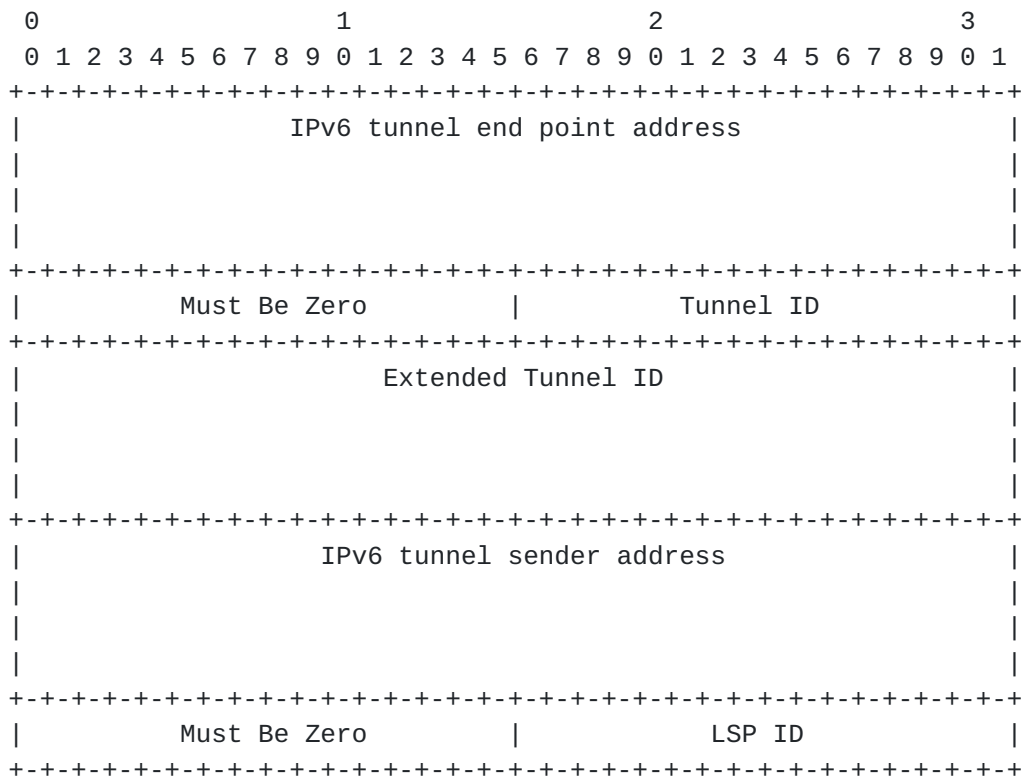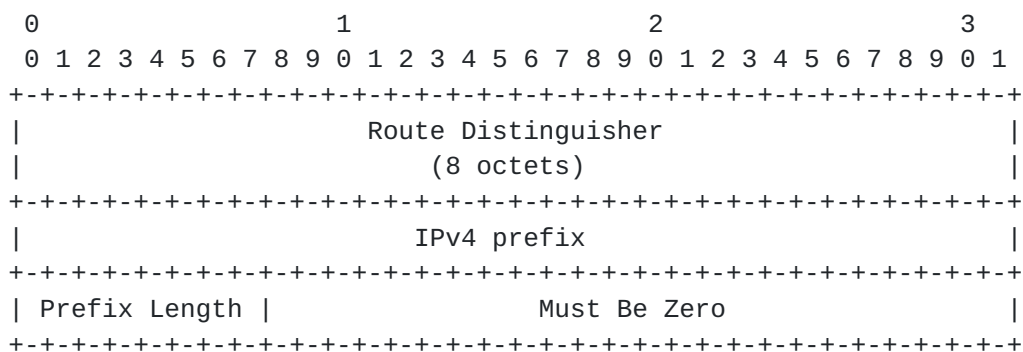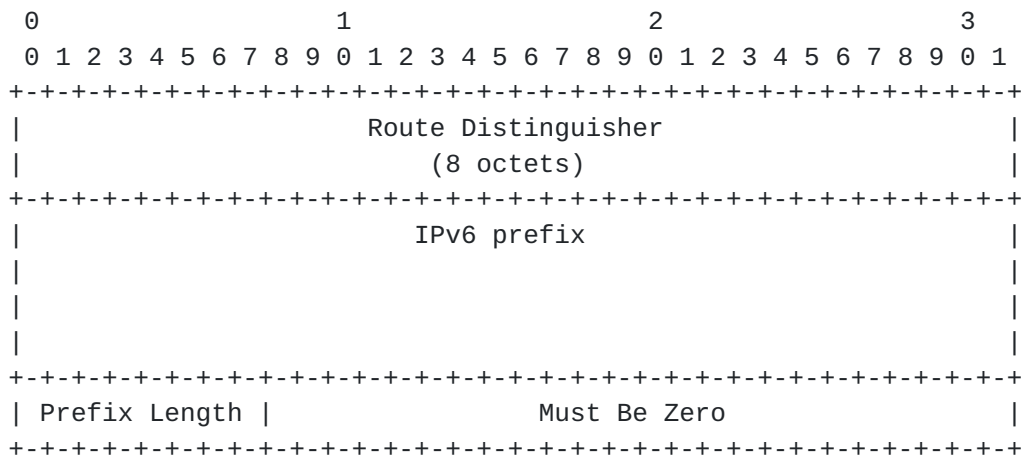
3.2.5. VPN IPv4 Prefix

The value field consists of the Route Distinguisher advertised with
the VPN IPv4 prefix, the IPv4 prefix (with trailing 0 bits to make 32
bits in all) and a prefix length, as follows:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      Route Distinguisher                      |
   |                           (8 octets)                          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                          IPv4 prefix                          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Prefix Length |                Must Be Zero                   |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

3.2.6. VPN IPv6 Prefix

The value field consists of the Route Distinguisher advertised with
the VPN IPv6 prefix, the IPv6 prefix (with trailing 0 bits to make
128 bits in all) and a prefix length, as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Route Distinguisher                      |
|                         (8 octets)                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        IPv6 prefix                           |
|                                                              |
|                                                              |
|                                                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Prefix Length |              Must Be Zero                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

[3.2.7](3.2.7). L2 VPN Endpoint

The value field consists of a Route Distinguisher (8 octets), the
sender (of the ping)'s VE ID (2 octets), the receiver's VE ID (2
octets), and an encapsulation type (2 octets), formatted as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Route Distinguisher                      |
|                         (8 octets)                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Sender's VE ID         |       Receiver's VE ID       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Encapsulation Type       |         Must Be Zero         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
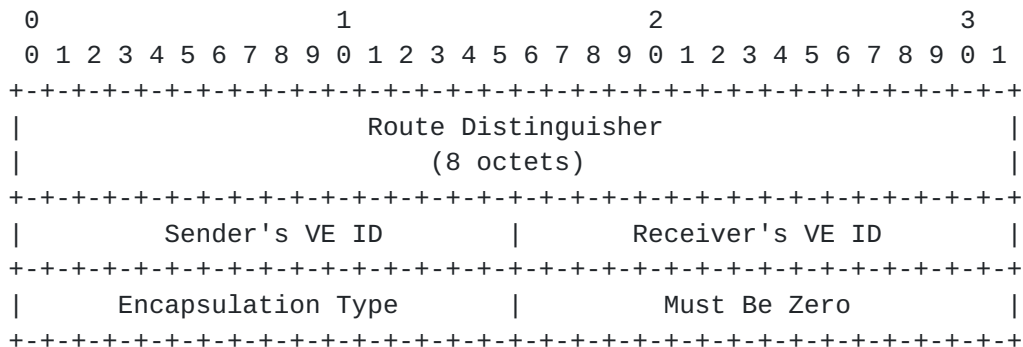
[3.2.8](3.2.8). FEC 128 Pseudowire (Deprecated)

The value field consists of the remote PE address (the destination
address of the targeted LDP session), a VC ID and an encapsulation
type, as follows:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      Remote PE Address                       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                            VC ID                             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Encapsulation Type        |         Must Be Zero         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

   This FEC will be deprecated, and is retained only for backward com-
   patibility. Implementations of LSP ping SHOULD accept and process
   this TLV, but SHOULD send LSP ping echo requests with the new TLV
   (see next section), unless explicitly asked by configuration to use
   the old TLV.

   An LSR receiving this TLV SHOULD use the source IP address of the LSP
   echo request to infer the Sender's PE Address.

3.2.9. FEC 128 Pseudowire (Current)

   The value field consists of the sender's PE address (the source
   address of the targeted LDP session), the remote PE address (the des-
   tination address of the targeted LDP session), a VC ID and an encap-
   sulation type, as follows:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      Sender's PE Address                     |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      Remote PE Address                       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                            VC ID                             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Encapsulation Type        |         Must Be Zero         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```


3.2.10. FEC 129 Pseudowire

   The Length of this TLV is 13 + AGI length + SAII length + TAII
   length. Padding is used to make the total length a multiple of 4;
   the length of the padding is not included in the Length field.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Sender's PE Address                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Remote PE Address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            PW Type            |  AGI Length  |  SAII Length   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  TAII Length  | AGI Value ... SAII Value ... TAII Value ...   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
. ...                                                           .
.                                                               .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| ...           | 0-3 octets of zero padding                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

[3.2.11](3.2.11). BGP Labeled IPv4 Prefix

The value field consists the IPv4 prefix (with trailing 0 bits to
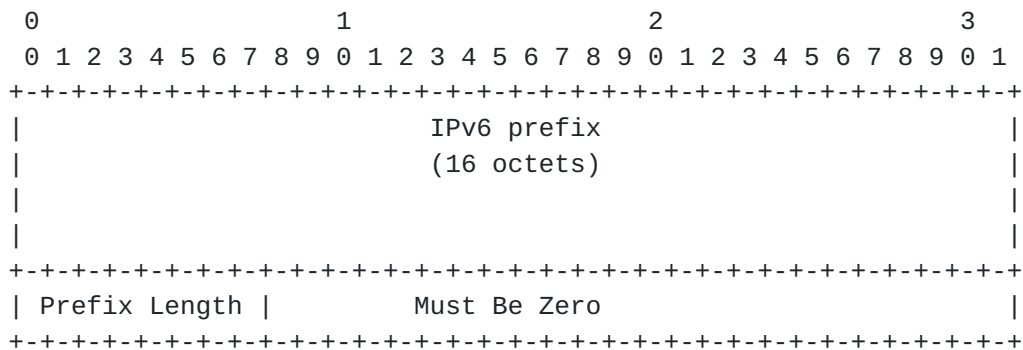make 32 bits in all), and the prefix length, as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          IPv4 Prefix                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Prefix Length |                Must Be Zero                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
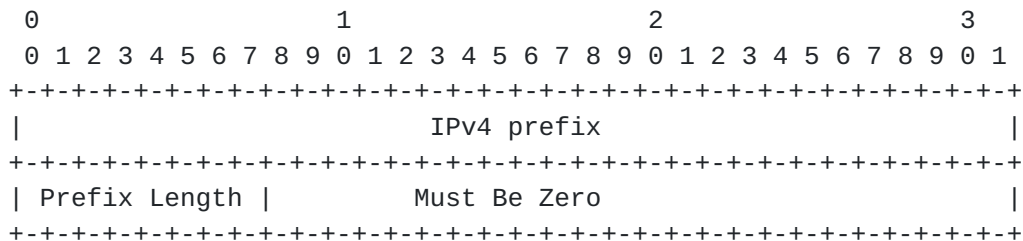
[3.2.12](3.2.12). BGP Labeled IPv6 Prefix

The value consists of sixteen octets of an IPv6 prefix followed by
one octet of prefix length in bits; the format is given below. The
IPv6 prefix is in network byte order; if the prefix is shorter than
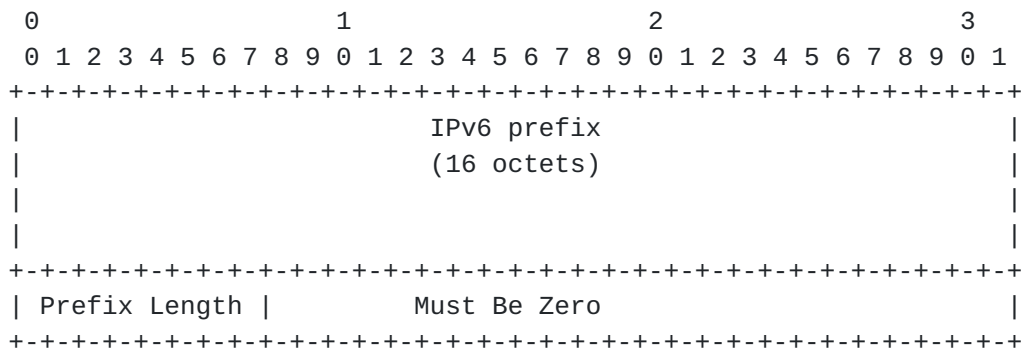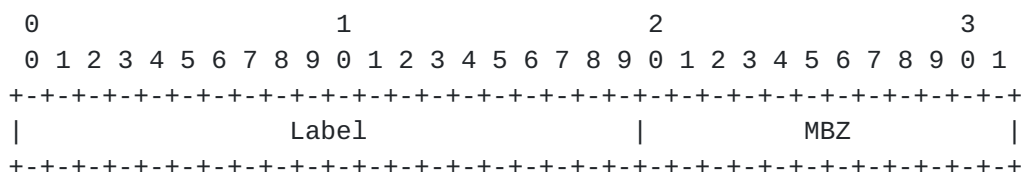128 bits, the trailing bits SHOULD be set to zero.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          IPv6 prefix                          |
|                          (16 octets)                          |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Prefix Length |           Must Be Zero                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

3.2.13. Generic IPv4 Prefix

The value consists of four octets of an IPv4 prefix followed by one
octet of prefix length in bits; the format is given below. The IPv4
prefix is in network byte order; if the prefix is shorter than 32
bits, trailing bits SHOULD be set to zero. This FEC is used if the
protocol advertising the label is unknown, or may change during the
course of the LSP. An example is an inter-AS LSP that may be sig-
naled by LDP in one AS, by RSVP-TE in another AS, and by BGP between
the ASes, such as is common for inter-AS VPNs.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          IPv4 prefix                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Prefix Length |           Must Be Zero                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

3.2.14. Generic IPv6 Prefix

The value consists of sixteen octets of an IPv6 prefix followed by
one octet of prefix length in bits; the format is given below. The
IPv6 prefix is in network byte order; if the prefix is shorter than
128 bits, the trailing bits SHOULD be set to zero.

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                         IPv6 prefix                          |
     |                         (16 octets)                          |
     |                                                              |
     |                                                              |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     | Prefix Length |           Must Be Zero                       |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

[3.2.15](3.2.15). Nil FEC

   At times labels from the reserved range, e.g. Router Alert and
   Explicit-null, may be added to the label stack for various diagnostic
   purposes such as influencing load-balancing. These labels may have
   no explicit FEC associated with them. The Nil FEC stack is defined
   to allow a Target FEC stack sub-TLV to be added to the target FEC
   stack to account for such labels so that proper validation can still
   be performed.

   The Length is 4. Labels are 20 bit values treated as numbers.
   stack.

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                    Label                  |       MBZ        |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

   Label is the actual label value inserted in the label stack; the MBZ
   fields MUST be zero when sent, and ignored on receipt.

[3.3](3.3). Downstream Mapping

   The Downstream Mapping object is a TLV which MAY be included in an
   echo request message. Only one Downstream Mapping object may appear
   in an echo request. The presence of a Downstream Mapping object is a
   request that Downstream Mapping objects be included in the echo
   reply. If the replying router is the destination of the FEC, then a
   Downstream Mapping TLV SHOULD NOT be included in the echo reply.
   Otherwise the replying router SHOULD include a Downstream Mapping
   object for each interface over which this FEC could be forwarded.
   For a more precise definition of the notion of "downstream", see the
   section named "Downstream".

The Length is K + M + 4*N octets, where M is the Multipath Length,
and N is the number of Downstream Labels. Values for K are found in
the description of Address Type below. The Value field of a Down-
stream Mapping has the following format:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |              MTU              | Address Type  |   DS Flags    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |            Downstream IP Address (4 or 16 octets)            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |         Downstream Interface Address (4 or 16 octets)        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Multipath Type| Depth Limit   |        Multipath Length      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   .                                                               .
   .                      (Multipath Information)                  .
   .                                                               .
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |               Downstream Label                |   Protocol    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   .                                                               .
   .                                                               .
   .                                                               .
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |               Downstream Label                |   Protocol    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
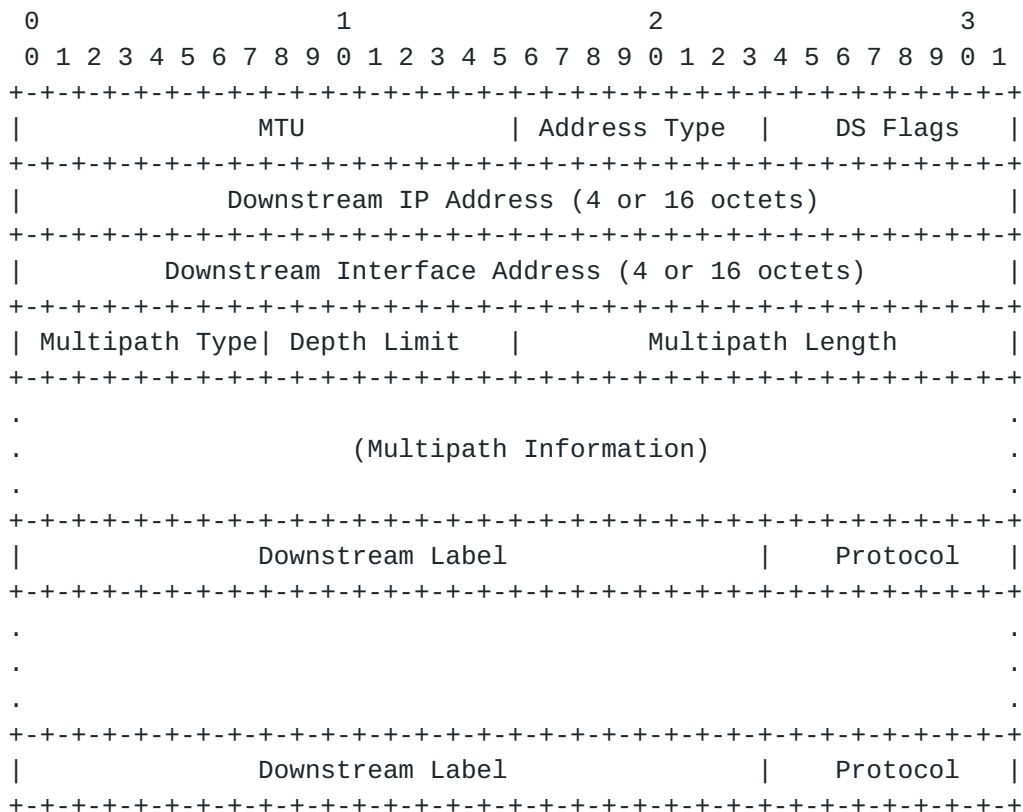
Maximum Transmission Unit (MTU)

   The MTU is the largest MPLS frame (including label stack) that fits
   on the interface to the Downstream LSR.

Address Type

   The Address Type indicates if the interface is numbered or unnum-
   bered. It also determines the length of the Downstream IP Address
   and Downstream Interface fields. The resulting total for the initial
   part of the TLV is listed in the table below as "K Octets". The
   Address Type is set to one of the following values:

Type #	Address Type	K Octets
1	IPv4 Numbered	16
2	IPv4 Unnumbered	16
3	IPv6 Numbered	40
4	IPv6 Unnumbered	28

DS Flags

    The DS Flags field is a bit vector with the following format:

         0 1 2 3 4 5 6 7
        +-+-+-+-+-+-+-+-+
        | Rsvd(MBZ) |I|N|
        +-+-+-+-+-+-+-+-+

    Two flags are defined currently, I and N. The remaining flags MUST
    be set to zero when sending, and ignored on receipt.

         Flag Name and Meaning
         ---- ----------------

            I Interface and Label Stack Object Request

               When this flag is set, it indicates that the replying
               router SHOULD include an Interface and Label Stack
               Object in the echo reply message

            N Treat as a Non-IP Packet

               Echo request messages will be used to diagnose non-IP
               flows. However, these messages are carried in IP
               packets. For a router which alters its ECMP algorithm
               based on the FEC or deep packet examination, this flag
               requests that the router treat this as it would if the
               determination of an IP payload had failed.

Downstream IP Address and Downstream Interface Address

    IPv4 addresses and and interface indices are encoded in 4 octets,
    IPv6 addresses are encoded in 16 octets.

    If the interface to the downstream LSR is numbered, then the Address
    Type MUST be set to IPv4 or IPv6, the Downstream IP Address MUST be
    set to either the downstream LSR's Router ID or the interface address

of the downstream LSR, and the Downstream Interface Address MUST be
set to the downstream LSR's interface address.

If the interface to the downstream LSR is unnumbered, the Address
Type MUST be IPv4 Unnumbered or IPv6 Unnumbered, the Downstream IP
Address MUST be the downstream LSR's Router ID, and the Downstream
Interface Address MUST be set to the index assigned by the upstream
LSR to the interface.

If an LSR does not know the IP address of its neighbor, then it MUST
set the Address Type to either IPv4 Unnumbered or IPv6 Unnumbered.
For IPv4 it must set the Downstream IP Address to 127.0.0.1, for IPv6
the address is set to 0::1. In both cases the interface index MUST
be set to 0. If an LSR receives an Echo Request packet with either
of these addresses in the Downstream IP Address field, this indicates
that it MUST bypass interface verification but continue with label
validation.

If the originator of an Echo Request packet wishes to obtain Down-
stream mapping information but does not know the expected label stack
then it SHOULD set the Address Type to either IPv4 Unnumbered or IPv6
Unnumbered. For IPv4 it MUST set the Downstream IP Address to
224.0.0.2, for IPv6 the address MUST be set to FF02::2. In both
cases the interface index MUST be set to 0. If an LSR receives an
Echo Request packet with the all-routers multicast address, then this
indicates that it MUST bypass both interface and label stack valida-
tion, but return Downstream Mapping TLVs using the information pro-
vided.

Multipath Type

   The following Mutipath Types are defined:

```
      Key   Type                  Multipath Information
      ---   ---------------       ---------------------
       0    no multipath          Empty (Multipath Length = 0)
       2    IP address            IP addresses
       4    IP address range      low/high address pairs
       8    Bit-masked IPv4       IP address prefix and bit mask
                address set
       9    Bit-masked label set  Label prefix and bit mask
```

   Type 0 indicates that all packets will be forwarded out this one
   interface.

   Types 2, 4, 8 and 9 specify that the supplied Multipath Information
   will serve to exercise this path.

Depth Limit

   The Depth Limit is applicable only to a label stack, and is the maxi-
   mum number of labels considered in the hash; this SHOULD be set to
   zero if unspecified or unlimited.

Multipath Length

   The length in octets of the Multipath Information.

Multipath Information

   Address or label values encoded according to the Multipath Type. See
   the next section below for encoding details.

Downstream Label(s)

   The set of labels in the label stack as it would have appeared if
   this router were forwarding the packet through this interface. Any
   Implicit Null labels are explicitly included. Labels are treated as
   numbers, i.e. they are right justified in the field.

   A Downstream Label is 24 bits, in the same format as an MPLS label
   minus the TTL field, i.e., the MSBit of the label is bit 0, the LSbit
   is bit 19, the EXP bits are bits 20-22, and bit 23 is the S bit. The
   replying router SHOULD fill in the EXP and S bits; the LSR receiving
   the echo reply MAY choose to ignore these bits.

Protocol

   The Protocol is taken from the following table:

        Protocol #   Signaling Protocol
        ----------   ------------------
               0   Unknown
               1   Static
               2   BGP
               3   LDP
               4   RSVP-TE

[3.3.1](3.3.1). **Multipath Information Encoding**

   The multipath information encodes labels or addresses which will
   exercise this path. The multipath information depends on the multi-
   path type. The contents of the field are shown in the table above.
   IP addresses are drawn from the range 127/8. Labels are treated as
   numbers, i.e. they are right justified in the field. For Type 4,
   ranges indicated by Address pairs MUST NOT overlap and MUST be in
   ascending sequence.

   Type 8 allows a denser encoding of IP address. The IPv4 prefix is
   formatted as a base IPv4 address with the non-prefix low order bits
   set to zero. The maximum prefix length is 27. Following the prefix
   is a mask of length 2^(32-prefix length) bits. Each bit set to one
   represents a valid address. The address is the base IPv4 address
   plus the position of the bit in the mask where the bits are numbered
   left to right beginning with zero. For example the IP addresses
   127.2.1.0, 127.2.1.5-127.2.1.15, and 127.2.1.20-127.2.1.29 would be
   encoded as follows:

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |0 1 1 1 1 1 1 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0|
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |1 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 1 1 1 1 1 1 1 1 1 0 0|
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

   Type 9 allows a denser encoding of Labels. The label prefix is for-
   matted as a base label value with the non-prefix low order bits set
   to zero. The maximum prefix (including leading zeros due to encod-
   ing) length is 27. Following the prefix is a mask of length
   2^(32-prefix length) bits. Each bit set to one represents a valid
   Label. The label is the base label plus the position of the bit in
   the mask where the bits are numbered left to right beginning with

zero. Label values of all the odd numbers between 1152 and 1279
would be encoded as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

If the received multipath information is non-null, the labels and IP
addresses MUST be picked from the set provided. If none of these
labels or addresses map to a particular downstream interface, then
for that interface, the type MUST be set to 0. If the received mul-
tipath information is null, (i.e. Multipath Length = 0, or for Types
8 and 9 a mask of all zeroes) the receiver the type MUST be set to 0.

For example, suppose LSR X at hop 10 has two downstream LSRs Y and Z
for the FEC in question. The received X could return Multipath Type
4, with low/high IP addresses of 127.1.1.1->127.1.1.255 for down-
stream LSR Y and 127.2.1.1->127.2.1.255 for downstream LSR Z. The
head end reflects this information to LSR Y. Y, which has three
downstream LSRs U, V and W, computes that 127.1.1.1->127.1.1.127
would go to U and 127.1.1.128-> 127.1.1.255 would go to V. Y would
then respond with 3 Downstream Mappings: to U, with Multipath Type 4
(127.1.1.1->127.1.1.127); to V, with Multipath Type 4
(127.1.1.127->127.1.1.255); and to W, with Multipath Type 0.

Note that computing multi-path information may impose a significant
processing burden on the receiver. A receiver MAY thus choose to
process a subset of the received prefixes. The sender, on receiving
a reply to a Downstream Map with partial information, SHOULD assume
that the prefixes missing in the reply were skipped by the receiver,
and MAY re-request information about them in a new echo request.

[3.3.2](3.3.2). Downstream Router and Interface

The notion of "downstream router" and "downstream interface" should
be explained. Consider an LSR X. If a packet that was originated
with TTL n>1 arrived with outermost label L and TTL=1 at LSR X, X
must be able to compute which LSRs could receive the packet if it was
originated with TTL=n+1, over which interface the request would
arrive and what label stack those LSRs would see. (It is outside the
scope of this document to specify how this computation is done.) The
set of these LSRs/interfaces are the downstream routers/interfaces
(and their corresponding labels) for X with respect to L. Each pair
of downstream router and interface requires a separate Downstream
Mapping to be added to the reply.

The case where X is the LSR originating the echo request is a special
case. X needs to figure out what LSRs would receive the MPLS echo
request for a given FEC Stack that X originates with TTL=1.

The set of downstream routers at X may be alternative paths (see the
discussion below on ECMP) or simultaneous paths (e.g., for MPLS mul-
ticast). In the former case, the Multipath Information is used as a
hint to the sender as to how it may influence the choice of these
alternatives.

[3.4](3.4). Pad TLV

The value part of the Pad TLV contains a variable number (>= 1) of
octets. The first octet takes values from the following table; all
the other octets (if any) are ignored. The receiver SHOULD verify
that the TLV is received in its entirety, but otherwise ignores the
contents of this TLV, apart from the first octet.

Value	Meaning
1	Drop Pad TLV from reply
2	Copy Pad TLV to reply
3-255	Reserved for future use

[3.5](3.5). Vendor Enterprise Code

   The Length is always 4; the value is the SMI Enterprise code, in net-
   work octet order, of the vendor with a Vendor Private extension to
   any of the fields in the fixed part of the message, in which case
   this TLV MUST be present. If none of the fields in the fixed part of
   the message have vendor private extensions, inclusion of this this
   TLV in is OPTIONAL. Vendor private ranges for Message Types, Reply
   Modes, and Return Codes have been defined. When any of these are
   used the Vendor Enterprise Code TLV MUST be included in the message.

[3.6](3.6). Interface and Label Stack

   The Interface and Label Stack TLV MAY be included in a reply message
   to report the interface on which the request message was received and
   the label stack which was on the packet when it was received. Only
   one such object may appear. The purpose of the object is to allow
   the upstream router to obtain the exact interface and label stack
   information as it appears at the replying LSR.

   The Length is K + 4*N octets, N is the number of labels in the Label
   Stack. Values for K are found in the description of Address Type
   below. The Value field of a Downstream Mapping has the following
   format:

   The value field of a Interface and Label Stack TLV has the following
   format:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Address Type  |              Must be Zero                     |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                   IP Address (4 or 16 octets)                 |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                    Interface (4 or 16 octets)                 |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                Downstream Interface Address                   |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   .                                                               .
   .                                                               .
   .                         Label Stack                           .
   .                                                               .
   .                                                               .
   .                                                               .
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Address Type

    The Address Type indicates if the interface is numbered or unnum-
    bered. It also determines the length of the Downstream IP Address
    and Downstream Interface fields. The resulting total for the initial
    part of the TLV is listed in the table below as "K Octets". The
    Address Type is set to one of the following values:

             Type #   Address Type   K Octets
             ------   ------------   --------
                  1   IPv4 Numbered   12
                  2   IPv4 Unnumbered   12
                  3   IPv6 Numbered   36
                  4   IPv6 Unnumbered   24

IP Address and Interface

    IPv4 addresses and and interface indices are encoded in 4 octets,
    IPv6 addresses are encoded in 16 octets.

    If the interface upon which the echo request message was received is
    numbered, then the Address Type MUST be set to IPv4 or IPv6, the IP
    Address MUST be set to either the LSR's Router ID or the interface
    address, and the Interface MUST be set to the interface address.

    If the interface unnumbered, the Address Type MUST be either IPv4
    Unnumbered or IPv6 Unnumbered, the IP Address MUST be the LSR's
    Router ID, and the Interface MUST be set to the index assigned to the
    interface.

Label Stack

    The label stack of the received echo request message. If any TTL
    values have been changed by this router, they SHOULD be restored.

[3.7](3.7). **Errored TLVs**

    The following TLV is a TLV which MAY be included in an echo reply to
    inform the sender of an echo request of Mandatory TLVs either not
    supported by an implementation, or parsed and found to be in error.

    The Value field contains the TLVs not understood encoded as sub-TLVs.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |               Type = 9            |             Length           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                            Value                              |
   .                                                              .
   .                                                              .
   .                                                              .
   |                                                              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

[3.8](3.8). Reply TOS Byte TLV

   This TLV MAY be used by the originator of the echo request to
   request
   that a echo reply be sent with the IP header TOS byte set to
   the value specified in the TLV. This TLV has a length of 4 with
   the following value field.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Reply-TOS Byte|                Must be zero                   |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

[4](4). Theory of Operation

An MPLS echo request is used to test a particular LSP. The LSP to be
tested is identified by the "FEC Stack"; for example, if the LSP was
set up via LDP, and is to an egress IP address of 10.1.1.1, the FEC
stack contains a single element, namely, an LDP IPv4 prefix sub-TLV
with value 10.1.1.1/32. If the LSP being tested is an RSVP LSP, the
FEC stack consists of a single element that captures the RSVP Session
and Sender Template which uniquely identifies the LSP.

FEC stacks can be more complex. For example, one may wish to test a
VPN IPv4 prefix of 10.1/8 that is tunneled over an LDP LSP with
egress 10.10.1.1. The FEC stack would then contain two sub-TLVs, the
bottom being a VPN IPv4 prefix, and the top being an LDP IPv4 prefix.
If the underlying (LDP) tunnel were not known, or was considered
irrelevant, the FEC stack could be a single element with just the VPN
IPv4 sub-TLV.

When an MPLS echo request is received, the receiver is expected to
verify that the control plane and data plane are both healthy (for

   the FEC stack being pinged), and that the two planes are in sync.
   The procedures for this are in section 4.4 below.

4.1. Dealing with Equal-Cost Multi-Path (ECMP)

   LSPs need not be simple point-to-point tunnels. Frequently, a single
   LSP may originate at several ingresses, and terminate at several
   egresses; this is very common with LDP LSPs. LSPs for a given FEC
   may also have multiple "next hops" at transit LSRs. At an ingress,
   there may also be several different LSPs to choose from to get to the
   desired endpoint. Finally, LSPs may have backup paths, detour paths
   and other alternative paths to take should the primary LSP go down.

   To deal with the last two first: it is assumed that the LSR sourcing
   MPLS echo requests can force the echo request into any desired LSP,
   so choosing among multiple LSPs at the ingress is not an issue. The
   problem of probing the various flavors of backup paths that will typ-
   ically not be used for forwarding data unless the primary LSP is down
   will not be addressed here.

   Since the actual LSP and path that a given packet may take may not be
   known a priori, it is useful if MPLS echo requests can exercise all
   possible paths. This, while desirable, may not be practical, because
   the algorithms that a given LSR uses to distribute packets over
   alternative paths may be proprietary.

   To achieve some degree of coverage of alternate paths, there is a
   certain latitude in choosing the destination IP address and source
   UDP port for an MPLS echo request. This is clearly not sufficient;
   in the case of traceroute, more latitude is offered by means of the
   Multipath Information of the Downstream Mapping TLV. This is used as
   follows. An ingress LSR periodically sends an MPLS traceroute mes-
   sage to determine whether there are multipaths for a given LSP. If
   so, each hop will provide some information how each of its downstream
   paths can be exercised. The ingress can then send MPLS echo requests
   that exercise these paths. If several transit LSRs have ECMP, the
   ingress may attempt to compose these to exercise all possible paths.
   However, full coverage may not be possible.

4.2. Testing LSPs That Are Used to Carry MPLS Payloads

To detect certain LSP breakages, it may be necessary to encapsulate
an MPLS echo request packet with at least one additional label when
testing LSPs that are used to carry MPLS payloads (such as LSPs used
to carry L2VPN and L3VPN traffic. For example, when testing LDP or
RSVP-TE LSPs, just sending an MPLS echo request packet may not detect
instances where the router immediately upstream of the destination of
the LSP ping may forward the MPLS echo request successfully over an
interface not configured to carry MPLS payloads because of the use of
penultimate hop popping. Since the receiving router has no means to
differentiate whether the IP packet was sent unlabeled or implicitly
labeled, the addition of labels shimmed above the MPLS echo request
(using the Nil FEC) will prevent a router from forwarding such a
packet out unlabeled interfaces.

4.3. Sending an MPLS Echo Request

An MPLS echo request is a (possibly) labeled UDP packet. The IP
header is set as follows: the source IP address is a routable address
of the sender; the destination IP address is a (randomly chosen)
address from 127/8; the IP TTL is set to 1. The source UDP port is
chosen by the sender; the destination UDP port is set to 3503
(assigned by IANA for MPLS echo requests). The Router Alert option
is set in the IP header.

If the echo request is labeled, one may (depending on what is being
pinged) set the TTL of the innermost label to 1, to prevent the ping
request going farther than it should. Examples of this include ping-
ing a VPN IPv4 or IPv6 prefix, an L2 VPN end point or a pseudowire.
This can also be accomplished by inserting a router alert label above
this label; however, this may lead to the undesired side effect that
MPLS echo requests take a different data path than actual data.

In "ping" mode (end-to-end connectivity check), the TTL in the outer-
most label is set to 255. In "traceroute" mode (fault isolation
mode), the TTL is set successively to 1, 2, ...

The sender chooses a Sender's Handle, and a Sequence Number. When
sending subsequent MPLS echo requests, the sender SHOULD increment
the sequence number by 1. However, a sender MAY choose to send a
group of echo requests with the same sequence number to improve the
chance of arrival of at least one packet with that sequence number.

The TimeStamp Sent is set to the time-of-day (in seconds and
microseconds) that the echo request is sent. The TimeStamp Received
is set to zero.

   An MPLS echo request MUST have a FEC Stack TLV. Also, the Reply Mode
   must be set to the desired reply mode; the Return Code and Subcode
   are set to zero. In the "traceroute" mode, the echo request SHOULD
   include a Downstream Mapping TLV.

[4.4](#). Receiving an MPLS Echo Request

   An LSR X that receives an MPLS echo request first parses the packet
   to ensure that it is a well-formed packet, and that the TLVs that are
   not marked "Ignore" are understood. If not, X SHOULD send an MPLS
   echo reply with the Return Code set to "Malformed echo request
   received" or "TLV not understood" (as appropriate), and the Subcode
   set to zero. In the latter case, the misunderstood TLVs (only) are
   included in the reply.

   If the echo request is good, X notes the interface I over which the
   echo was received, and the label stack with which it came.

   For reporting purposes the bottom of stack is considered to be stack-
   depth of 1. This is to establish an absolute reference for the case
   where the stack may have more labels than are in the FEC stack. Fur-
   ther, in all the error codes listed in this document a stack-depth of
   0 means "no value specified". This allows compatibility with exist-
   ing implementations which do not use the Return Subcode field.

   X employs two variables, called FEC-stack-depth and Label-stack-
   depth. X sets Label-stack-depth to the number of labels in the
   received label stack. If the label-stack-depth is 0, assume there is
   one implicit null label and set label-stack-depth to 1. FEC-stack-
   depth is used later and need not be initialized. Processing now con-
   tinues with the following steps:

   Label_Validation:

     If the label at Label-stack-depth is valid, goto Label_Operation.
     If not, set Best-return-code to 11, "No label entry at stack-depth"
     and Best-return-subcode to Label-stack-depth. Goto
     Send_Reply_Packet.

   Label_Operation:

     Switch on label operation.

     Case: Pop and Continue Processing (Note: this includes
             Explicit_Null and Router_Alert)

If Label-stack-depth is greater than 1, decrement Label-stack-
depth and goto Label_Validation. Otherwise, set FEC-stack-depth
to 1, set Best-return-code to 3 "Replying router is an egress for
the FEC at stack depth", set Best-return-subcode to 1 and goto
Egress_Processing.

Case: Swap or Pop and Switch based on Popped Label

If the label operation is either swap or pop and switch based on
the popped label, Best-return-code to 8, "Label switched at
stack-depth" and Best-return-subcode to Label-stack-depth.

If a Downstream Mapping TLV is present, a Downstream mapping TLVs
SHOULD be created for each multipath.

Determine the output interface. If it is not valid to forward a
labeled packet on this interface, set Best-return-code to Return
Code 9, "Label switched but no MPLS forwarding at stack-depth"
and set Best-return-subcode to Label-stack-depth and goto
Send_Reply_Packet. (Note: this return code is set even if Label-
stack-depth is one.)

If no Downstream Mapping TLV is present, or the Downstream IP
Address is set to the All-Routers multicast address goto
Send_Reply_Packet.

Verify that the IP address, interface address and label stack
match the received interface and label stack. If the IP address
is either 127.0.0.1 or 0::1 bypass the interface check, and set
Best-return-code to 6, "Upstream Interface Index Unknown". For
any other error, set Best-return-code to 5, "Downstream Mapping
Mis-match". For either error, an Interface and Label Stack TLV
SHOULD be created. If Best-return-code equals 5, goto
Send_Reply_Packet.

If the "Validate FEC Stack" flag is not set, goto
Send_Reply_Packet.

Locate the label at Label-stack-depth in the Downstream Labels by
counting from the bottom of the stack, skipping over, but count-
ing Implicit Null labels and set FEC-stack-depth to that depth.
(Note: If the Downstream Labels contain one or more Implicit Null
labels, this may be at a depth greater than Label-stack-depth.)

If the depth of the FEC stack is greater than or equal to FEC-
stack-depth, Perform FEC Checking. If FEC-status is 2, set Best-
return-code to 10, "Mapping for this FEC is not the given label
at stack-depth".

       If the return code is 1 set Best-return-code to FEC-return-code
       and Best-return-subcode to FEC-stack-depth.

       Goto Send_Reply_Packet.

   Egress_Processing:

     If no Downstream Mapping TLV is present, goto Egress_FEC_Valida-
     tion.

     Verify that the IP address, interface address and label stack match
     the received interface and label stack. If not, set Best-return-
     code to 5, "Downstream Mapping Mis-match". A Received Interface
     and Label Stack TLV SHOULD be created. Goto Send_Reply_Packet.

   Egress_FEC_Validation:

     Perform FEC checking. If FEC-status is 1, set Best-return-code
     to FEC-code and Best-return-subcode to FEC-stack-depth. Goto
     Send_Reply_Packet.

     Increment FEC-stack-depth. If FEC-stack-depth is greater than
     the number of FECs in the FEC-stack, goto Send_Reply_Packet. If
     FEC-status is 0, increment Label-stack-depth. Goto
     Egress_FEC_Validation.

   Send_Reply_Packet:

     Send an MPLS echo reply with a Return Code of Best-return-code,
     and a Return Subcode of Best-return-subcode. Include any TLVs
     created during the above process. The procedures for sending the
     echo reply are found in the next subsection below.

   FEC_Checking:

     This routine accepts a FEC, Label, and Interface. It returns two
     values, FEC-status and FEC-return-code, both of which are
     initialized to 0.

     If the FEC is the Nil FEC, check that Label is either
     Explicit_Null or Router_Alert. If so return. Else
     set FEC-return-code to 10, "Mapping for this FEC is not the given
     label at stack-depth". Set FEC-status to 1 and return.

   Check that the label mapping for FEC. If no mapping exists, set
   FEC-return-code to Return 4, "Replying router has no mapping for
   the FEC at stack-depth". Set FEC-status to 1. Return.

   If the label mapping for FEC is Implicit Null, set FEC-status to
   2. Goto Check_Protocol.

   If the label mapping for FEC is Label, goto Check_Protocol. Else
   set FEC-return-code to 10, "Mapping for this FEC is not the given
   label at stack-depth". Set FEC-status to 1 and return.

Check_Protocol:

   Check what protocol would be used to advertise FEC. If it can be
   determined that no protocol associated with interface I would
   have advertised a FEC of that FEC-Type, set FEC-return-code to
   12, "Protocol not associated with interface at FEC stack-depth".
   Set FEC-status to 1. Return.

4.5. Sending an MPLS Echo Reply

An MPLS echo reply is a UDP packet. It MUST ONLY be sent in response
to an MPLS echo request. The source IP address is a routable address
of the replier; the source port is the well-known UDP port for LSP
ping. The destination IP address and UDP port are copied from the
source IP address and UDP port of the echo request. The IP TTL is
set to 255. If the Reply Mode in the echo request is "Reply via an
IPv4 UDP packet with Router Alert", then the IP header MUST contain
the Router Alert IP option. If the reply is sent over an LSP, the
topmost label MUST in this case be the Router Alert label (1) (see
[LABEL-STACK]).

The format of the echo reply is the same as the echo request. The
Sender's Handle, the Sequence Number and TimeStamp Sent are copied
from the echo request; the TimeStamp Received is set to the time-of-
day that the echo request is received (note that this information is
most useful if the time-of-day clocks on the requester and the
replier are synchronized). The FEC Stack TLV from the echo request
MAY be copied to the reply.

The replier MUST fill in the Return Code and Subcode, as determined
in the previous subsection.

If the echo request contains a Pad TLV, the replier MUST interpret
the first octet for instructions regarding how to reply.

If the replying router is the destination of the FEC, then Downstream
Mapping TLVs SHOULD NOT be included in the echo reply.

If the echo request contains a Downstream Mapping TLV, and the reply-
ing router is not the destination of the FEC, the replier SHOULD com-
pute its downstream routers and corresponding labels for the incoming
label, and add Downstream Mapping TLVs for each one to the echo reply
it sends back.

If the Downstream Mapping TLV contains multipath information requir-
ing more processing than the receiving router is willing to perform,
the responding router MAY choose to respond with only a subset of
multipaths contained in the echo request Downstream Map. (Note: The
originator of the echo request MAY send another echo request with the
multipath information that was not included in the reply.)

[4.6](4.6). Receiving an MPLS Echo Reply

An LSR X should only receive an MPLS echo reply in response to an
MPLS echo request that it sent. Thus, on receipt of an MPLS echo
reply, X should parse the packet to assure that it is well-formed,
then attempt to match up the echo reply with an echo request that it
had previously sent, using the destination UDP port and the Sender's
Handle. If no match is found, then X jettisons the echo reply; oth-
erwise, it checks the Sequence Number to see if it matches. Gaps in
the Sequence Number MAY be logged and SHOULD be counted. Once an
echo reply is received for a given Sequence Number (for a given UDP
port and Handle), the Sequence Number for subsequent echo requests
for that UDP port and Handle SHOULD be incremented.

If the echo reply contains Downstream Mappings, and X wishes to
traceroute further, it SHOULD copy the Downstream Mapping(s) into its
next echo request(s) (with TTL incremented by one).

[4.7](4.7). Issue with VPN IPv4 and IPv6 Prefixes

Typically, a LSP ping for a VPN IPv4 or IPv6 prefix is sent with a
label stack of depth greater than 1, with the innermost label having
a TTL of 1. This is to terminate the ping at the egress PE, before
it gets sent to the customer device. However, under certain circum-
stances, the label stack can shrink to a single label before the ping
hits the egress PE; this will result in the ping terminating prema-
turely. One such scenario is a multi-AS Carrier's Carrier VPN.

To get around this problem, one approach is for the LSR that receives
such a ping to realize that the ping terminated prematurely, and send

back error code 13. In that case, the initiating LSR can retry the
ping after incrementing the TTL on the VPN label. In this fashion,
the ingress LSR will sequentially try TTL values until it finds one
that allows the VPN ping to reach the egress PE.

4.8. Non-compliant Routers

If the egress for the FEC Stack being pinged does not support MPLS
ping, then no reply will be sent, resulting in possible "false nega-
tives". If in "traceroute" mode, a transit LSR does not support LSP
ping, then no reply will be forthcoming from that LSR for some TTL,
say n. The LSR originating the echo request SHOULD try sending the
echo request with TTL=n+1, n+2, ..., n+k to probe LSRs further down
the path. In such a case, the echo request for TTL > n SHOULD be
sent with Downstream Mapping TLV "Downstream IP Address" field set to
the ALLROUTERs multicast address until a reply is received with a
Downstream Mapping TLV. The Label Stack MAY be omitted from the
Downstream Mapping TLV. Further the "Validate FEC Stack" flag SHOULD
NOT be set until an echo reply packet with a Downstream Mapping TLV
is received.

5. References

Normative References

    [IANA]   Narten, T. and H. Alvestrand, "Guidelines for IANA
                  Considerations", BCP 26, RFC 2434, October 1998.

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

    [LABEL-STACK] Rosen, E., et al, "MPLS Label Stack Encoding",
                  RFC 3032, January 2001.

Informative References

    [ICMP]   Postel, J., "Internet Control Message Protocol",
                  RFC 792.

    [LDP]   Andersson, L., et al, "LDP Specification", RFC 3036,
                  January 2001.

[6](6). Security Considerations

   There are at least two approaches to attacking LSRs using the mecha-
   nisms defined here. One is a Denial of Service attack, by sending
   MPLS echo requests/replies to LSRs and thereby increasing their work-
   load. The other is obfuscating the state of the MPLS data plane
   liveness by spoofing, hijacking, replaying or otherwise tampering
   with MPLS echo requests and replies.

   Authentication will help reduce the number of seemingly valid MPLS
   echo requests, and thus cut down the Denial of Service attacks;
   beyond that, each LSR must protect itself. To avoid potential Denial
   of Service attacks, it is RECOMMENDED that implementations regulate
   the LSP ping traffic going to the control plane. A rate limiter
   SHOULD be applied to the well-known UDP port defined below.

   Authentication sufficiently addresses spoofing, replay and most tam-
   pering attacks; one hopes to use some mechanism devised or suggested
   by the RPSec WG. It is not clear how to prevent hijacking (non-
   delivery) of echo requests or replies; however, if these messages are
   indeed hijacked, LSP ping will report that the data plane isn't work-
   ing as it should.

   It doesn't seem vital (at this point) to secure the data carried in
   MPLS echo requests and replies, although knowledge of the state of
   the MPLS data plane may be considered confidential by some.

[7](7). IANA Considerations

   The TCP and UDP port number 3503 has been allocated by IANA for LSP
   echo requests and replies.

   The following sections detail the new name spaces to be managed by
   IANA. For each of these name spaces, the space is divided into
   assignment ranges; the following terms are used in describing the
   procedures by which IANA allocates values: "Standards Action" (as
   defined in [IANA]); "Expert Review" and "Vendor Private Use".

   Values from "Expert Review" ranges MUST be registered with IANA, and
   MUST be accompanied by an Experimental RFC that describes the format
   and procedures for using the code point; the actual assignment is
   made during the IANA actions for the RFC.

   Values from "Vendor Private" ranges MUST NOT be registered with IANA;
   however, the message MUST contain an enterprise code as registered
   with the IANA SMI Network Management Private Enterprise Codes. For
   each name space that has a Vendor Private range, it must be specified

where exactly the SMI Enterprise Code resides; see below for exam-
ples. In this way, several enterprises (vendors) can use the same
code point without fear of collision.

7.1. Message Types, Reply Modes, Return Codes

It is requested that IANA maintain registries for Message Types,
Reply Modes, and Return Codes. Each of these can take values in the
range 0-255. Assignments in the range 0-191 are via Standards
Action; assignments in the range 192-251 are made via Expert Review;
values in the range 252-255 are for Vendor Private Use, and MUST NOT
be allocated.

If any of these fields fall in the Vendor Private range, a top-level
Vendor Enterprise Code TLV MUST be present in the message.

Message Types defined in this document are:

```
    Value    Meaning
    -----    -------
        1    MPLS Echo Request
        2    MPLS Echo Reply
```

Reply Modes defined in this document are:

```
    Value    Meaning
    -----    -------
        1    Do not reply
        2    Reply via an IPv4/IPv6 UDP packet
        3    Reply via an IPv4/IPv6 UDP packet with Router Alert
        4    Reply via application level control channel
```

Return Codes defined in this document are listed in section 3.1.

7.2. TLVs

It is requested that IANA maintain a registry for the Type field of
top-level TLVs as well as for any associated sub-TLVs. Note the
meaning of a sub-TLV is scoped by the TLV. The valid range for each
of these is 0-65535. Assignments in the range 0-16383 and
32768-49161 are made via Standards Action as defined in [IANA];
assignments in the range 16384-31743 and 49162-64511 are made via
Expert Review (see below); values in the range 31744-32746 and
64512-65535 are for Vendor Private Use, and MUST NOT be allocated.

If a TLV or sub-TLV has a Type that falls in the range for Vendor

Private Use, the Length MUST be at least 4, and the first four octets
MUST be that vendor's SMI Enterprise Code, in network octet order.
The rest of the Value field is private to the vendor.

TLVs and sub-TLVs defined in this document are:

Type	Sub-Type	Value Field
1		Target FEC Stack
	1	LDP IPv4 prefix
	2	LDP IPv6 prefix
	3	RSVP IPv4 LSP
	4	RSVP IPv6 LSP
	5	Not Assigned
	6	VPN IPv4 prefix
	7	VPN IPv6 prefix
	8	L2 VPN endpoint
	9	"FEC 128" Pseudowire (Deprecated)
	10	"FEC 128" Pseudowire
	11	"FEC 129" Pseudowire
	12	BGP labeled IPv4 prefix
	13	BGP labeled IPv6 prefix
	14	Generic IPv4 prefix
	15	Generic IPv6 prefix
	16	Nil FEC
2		Downstream Mapping
3		Pad
4		Not Assigned
5		Vendor Enterprise Code
6		Not Assigned
7		Interface and Label Stack
8		Not Assigned
9		Errored TLVs
	Any value	The TLV not understood
10		Reply TOS Byte

[8](8). Acknowledgments

This document is the outcome of many discussions among many people,
that include Manoj Leelanivas, Paul Traina, Yakov Rekhter, Der-Hwa
Gan, Brook Bailey, Eric Rosen, Ina Minei, Shivani Aggarwal and Vanson
Lim.

The description of the Multipath Information sub-field of the Down-
stream Mapping TLV was adapted from text suggested by Curtis Vil-
lamizar.

Authors' Address

   Kireeti Kompella
   Juniper Networks
   1194 N.Mathilda Ave
   Sunnyvale, CA 94089
   Email: kireeti@juniper.net

   George Swallow
   Cisco Systems
   1414 Massachusetts Ave,
   Boxborough, MA 01719
   Phone: +1 978 936 1398
   Email: swallow@cisco.com

Expiration Date

   November 2005