Network Working Group
Internet Draft
Expiration Date: July 2000

Lou Berger
LabN Consulting, LLC

Jason Jeffords
Integral Access Inc.

January 2000

MPLS/IP Header Compression

draft-berger-mpls-hdr-comp-00.txt

Status of this Memo

Abstract

This document describes a method for compressing the headers of IP datagrams that are being transported over MPLS. This work extends the existing IP and IP/UDP/RTP header compression techniques, as defined in [RFC2507] and [RFC2508], to operate over and to compress MPLS label stack entries.

Changes from previous version:

o Initial draft.

1. Introduction

IP and IP/UDP/RTP header compression, which is defined in [RFC2507] and [RFC2508], reduces header overhead and is particularly useful on lower speed links. The current header compression definition relies on IP datagrams being carried directly over a link layer protocol, such as PPP [STD51]. With the introduction of MPLS [LABELS], one or more MPLS headers, which are called label stack entries, may be present between the IP and link layer headers. Since the header following the link layer header is no longer IP, the existing compression techniques will not operate. Clearly delivering header compression when using MPLS on lower speed links is desirable.

This document presents one method for providing header compression while running over MPLS. The presented method incrementally builds on the header compression techniques defined in [RFC2507] and [RFC2508]. It makes use of the same basic mechanisms and continues to provide compression on a link-by-link basis. It preserves the ability to compress all headers, including multiple MPLS label stack entries, into the same space when MPLS EXP bits remain constant. This is 2-4 bytes for MPLS/IP/UDP/RTP header compression. It also co-exists with standard IP, IP/TCP and IP/UDP/RTP header compression running on the same link.

Familiarity with [RFC2507], [RFC2508] and [RFC1144] is encouraged as the material presented in those documents is not repeated in this document and their principals and techniques are leveraged.

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.

2. Compression Method Overview

The defined compression method reuses the compression techniques and formats defined for IP and IP/UDP/RTP compression. IP and IP/UDP/RTP compression relies on many header fields not changing per packet over the life of the session, or some fields differing from packet to packet by only a constant value. By maintaining both the uncompressed header and the first-order differences in the session state shared between the compressor and decompressor, all that must be communicated is an indication that the second-order difference was

zero. The decompressor can then reconstruct the original header without any loss of information simply by using the packet length indicated by the link-level protocol and by adding the first-order differences to the saved uncompressed header as each compressed packet is received.

The same compression techniques can be used on MPLS label stack entries. All fields within a stack entry other than the EXP field will typically remain constant over the life of a session. When a label stack is used, i.e., there is more than one stack entry, the number of stack entries and the label values of each will also remain constant. To support MPLS header compression, the number of stack entries and each entry's label is added to the session context.

As with the previous techniques, the compressor and decompressor must maintain state per session context. Likewise, multiple session contexts should be maintained. In addition to the new MPLS related information, the session context contains the combination of the IP source and destination addresses, the TCP or UDP source and destination ports, and for RTP the SSRC field. As with IP/UDP/RTP compression, a compressor implementation might use a hash function on these fields to index a table of stored session contexts. The compressed packet continues to carry a small integer, called the session context identifier or CID, to indicate in which session context that packet should be interpreted. Packets containing full headers and a CID are transmitted to initiate compression and to keep the decompressor synchronized. The decompressor can use the CID to index its table of stored session contexts directly. The CID together with any header fields that differ by a non-constant value are all that must be conveyed in order for the decompressor to reconstruct the full original MPLS, IP, TCP, UDP and RTP headers. Note that the existing TCP and non-TCP CID spaces are reused for MPLS encapsulated traffic.

If not explicitly mentioned, all the conditions and restrictions documented in [RFC2507] and [RFC2508] apply to this document. For example, this includes the restrictions on handling of fragmented IP packets, and the need for most compressor implementations to maintain a "negative cache" of packet streams that have failed to compress as RTP packets.

Other methods for supporting IP and IP/UDP/RTP header compression over MPLS are possible and some where considered. One alternative worth noting is adapting the existing header compression techniques to operate directly over MPLS without any MPLS header compression. This method was discounted for a couple of reasons. The first was that MPLS headers do not typically carry a packet type code. Such codes could be carried by assigning special meaning to the EXP field

or even the label field. It was decided that such assignment was undesirable due to possible collision with Diff-Serv usage of the EXP field and not wanting to subdivide the label field. The second and more important reason for not adapting compression to operate over MPLS was simply to gain extra efficiency by enabling the compression of the MPLS label stack.

3. MPLS/IP Header Compression

MPLS headers, i.e., label stack entries, are carried between the IP header and the link layer header. The number of stack entries and the label values of each will remain constant through the life of a session. Data flows that have different number of stack entries or different label values are always considered to be different sessions. The only part of a stack entry that may change during a session is the EXP field, and this is only the case for some sessions.

To support a session that uses MPLS headers, the compressor and decompressor must synchronize, on a per MPLS session basis, on the number of stack entries and on the contents of each entry. Once a compressor and decompressor are synchronized on the MPLS headers, the headers need not be resent. For the cases where the EXP field my regularly change, the changed EXP fields must be conveyed per packet. When multiple stack entries are present, more than one EXP field may change in a label stack.

Other than the initial synchronization of MPLS information and possible EXP field support, the previously defined header compression techniques can be reused with little or no modification. The primary modifications required are to session context information, and the communication of uncompressed MPLS headers.

As with IP and IP/UDP/RTP compression, there is a separate session context for each MPLS/IP packet stream. The number of session contexts to be maintained MAY continue to be negotiated between the compressor and decompressor. The support of MPLS/IP compression MUST be negotiated or configured. The following information is added to the shared information in each context, as defined in [RFC2507] and [RFC2508]:

- The number of stack entries.
- The full MPLS stack.
- Whether EXP changes are being supported. This may change over the life of a session.

To communicate this new shared information, a new packet format is defined:

- FULL_MPLS_HEADER - communicates the uncompressed MPLS stack plus any following headers and data to establish the uncompressed header state in the decompressor for a particular context. The FULL_MPLS_HEADER packet also carries the 8- or 16-bit session context identifier and other previously defined fields to establish synchronization between the compressor and decompressor.

Also to eliminate the possibility of interpreting an MPLS session as non-MPLS sessions in the face of certain loss conditions, the following packet format is defined:

- COMPRESSED_MPLS - indicates a packet with a compressed MPLS header. This packet contains a full IP header. It is used in place of the FULL_HEADER packet.

Assignments of numeric codes for these packet formats for PPP [STD51] are to be made by the Internet Assigned Numbers Authority.

3.1. FULL_MPLS_HEADER Packet Format

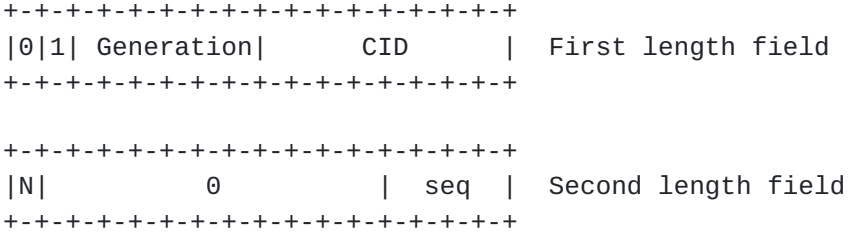
The FULL_MPLS_HEADER packet is used to associate a compression context ID with full MPLS, IP, TCP or UDP and RTP headers, or to update the contents of some of those headers. It is only used when one or more MPLS stack entries are preset.

As with the existing IP and IP/UDP/RTP compression, the format of the FULL_MPLS_HEADER packet is the same as that of the original packet. The FULL_MPLS_HEADER packet differs from the corresponding normal MPLS/IPv4 or MPLS/IPv6 packet in that it must also carry the compression context ID and other previously defined compression related fields. The compression related information in a FULL_MPLS_HEADER packet differs from the information defined in Section 5.3.2 of [RFC2507] and Section 3.3.1 of [RFC2508] in that it also contains an MPLS related flag. The flag, called the NoEXP or "N" bit, indicates when the EXP bits will NOT be conveyed in every compressed packet. The position of the compression related information is unchanged and is specified in Section 5.3.2 of [RFC2507].

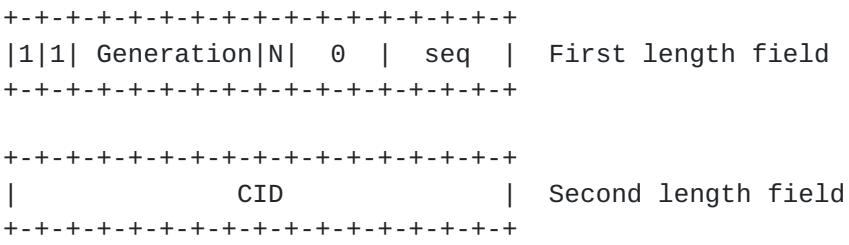
The definition of the compression related information carried in a FULL_MPLS_HEADER packet is:

For MPLS/IP/UDP sessions [RFC 2508, Section 3.3.1]

For 8-bit context ID:

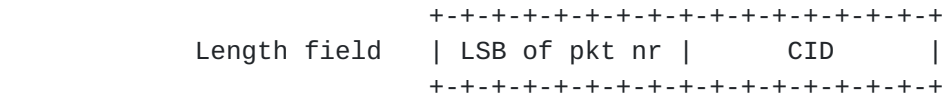


For 16-bit context ID:

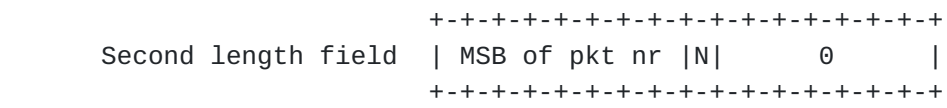


For MPLS/IP/TCP sessions [RFC 2507, Section 5.3.1]

Use of first length field:

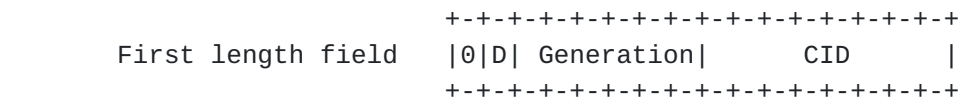


Use of second length field if available:

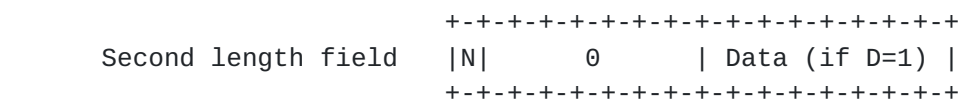


For MPLS/non-TCP sessions [RFC 2507, Section 5.3.1]

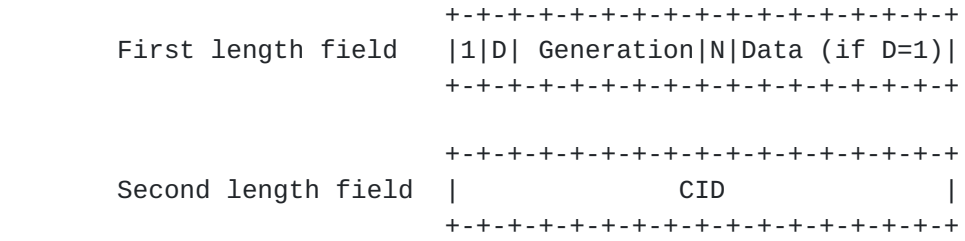
For non-TCP headers with 8-bit CID:



Use of second length field if available:



Full non-TCP headers with 16-bit CID:



The only new field introduced in the above formats is the N-bit. The N-bit, when set, indicates that the EXP fields in all present MPLS label stack entries will remain constant. When not set, i.e., is zero (0), one or more EXP fields may change on a packet-by-packet basis. In this case, a new EXP Compression field is included in all compressor to decompressor packets defined in [RFC2507] and [RFC2508]. See Section 3.5 for more details. Note that in the 8-bit CID cases the N-bit is carried in the second length field. When the second length field is not available, the N-Bit is not carried and and the use of the EXP Compression field is implied.

With the exception of the N-Bit all fields have the same meaning and use as with existing IP and IP/UDP/RTP compression. Processing of a MPLS_FULL_HEADER parallels the processing of a FULL_HEADER packet. As with the FULL_HEADER packet, a receiver of a MPLS_FULL_HEADER packet stores the complete set of headers into the context selected by the context ID. The other compression related information is also stored in the context, thereby resynchronizing the decompressor to the compressor.

MPLS_FULL_HEADER packets are sent for new sessions, after the receipt of a CONTEXT_STATE message indicating that synchronization has been lost for the associated CID, and periodically as called for in [RFC2507] and [RFC2508]. MPLS_FULL_HEADER packets are also sent to

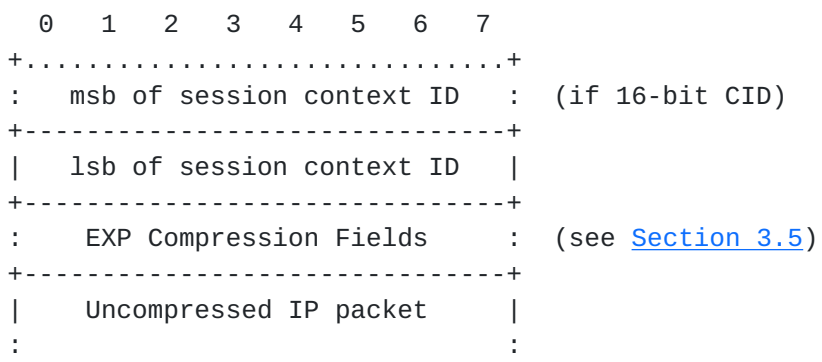
indicate a change in EXP field handling.

The compressor and decompressor need to agree on the number of MPLS label stack entries that may be present before an IP header. The compressor and decompressor SHOULD negotiate the maximum number allowed. If no maximum number of MPLS headers is negotiated or configured, a default of one (1) SHALL be used. Packets containing more than the maximum number of MPLS headers MUST not be compressed.

3.2. COMPRESSED_MPLS Packet Format

The COMPRESSED_MPLS packet carries a compressed MPLS label stack and all other headers uncompressed. It is used to cover the cases where a full IP and TCP or UDP header must be sent.

The format of a COMPRESSED_MPLS packet is:



3.3. Baseline Operation

Under normal conditions the number of MPLS label stack entries and the contents of each entry is expected to remain constant over the life of a session. (Section 3.5 covers the case where the EXP field is expected to change on a per packet basis.) Operation in this mode is selected by the N-bit, for more details see section 3.4. In this mode, all MPLS headers will be transmitted in MPLS_FULL_HEADER packets and stored by the decompressor. Once the MPLS headers are transmitted and stored by the decompressor, they no longer need to be transmitted. This means that after a MPLS_FULL_HEADER packet is sent for a particular session, the standard IP and IP/UDP/RTP compression packet formats and associated processing may be used without any modifications.

While the standard packet formats are used, the decompressor must of course be extended to reconstructs the original MPLS headers for all MPLS session. When reconstructing the headers of an MPLS/IP packet

received in any compressed packet type, the decompressor MUST perform the same context validation checks as described in Section 10.2 of [RFC2507] and Section 3.3.5 of [RFC2508]. If the checks fail, a corresponding CONTEXT_STATE packet MUST be sent. If the checks pass the decompressor adds the MPLS headers associated with the CID.

There is one standard packet type that must be handled differently. For sessions not associated with MPLS, the FULL_HEADER packet is used to identify and reset the session context. When a session is identified as using MPLS headers, via a MPLS_FULL_HEADER packet, the FULL_HEADER packet must be handled differently. For MPLS sessions, the FULL_HEADER packet is not used. If a FULL_HEADER packet is received with a CID matching an MPLS session, the receiver must assume that the CID is being reused for a new non-MPLS session and it MUST reset all session context based on the received FULL_HEADER packet.

When a CONTEXT_STATE packet is received for an MPLS session, a MPLS_FULL_HEADER packet MUST be sent rather than a FULL_HEADER packet. Note that per [RFC2508] and [RFC2507] generating a packet is not always required in response to a received a CONTEXT_STATE packet.

3.4. Setting the N-bit

The N-bit is carried in MPLS_FULL_HEADER packets and controls whether EXP bits are carried in compressed packets. When the EXP fields in all present MPLS label stack entries are expected to remain constant on a packet-by-packet basis, the N-bit SHOULD be set. Values used in the MPLS EXP field typically depend on whether Differentiated Services [DIFFSERV] is being supported. When Diff-Serv is not being used, the EXP field will typically remain constant. When Diff-Serv is being supported, the EXP field may change packet-by-packet for some applications and remain constant for others. Because of these different cases, it will generally be difficult to identify which MPLS sessions will be regularly changing EXP fields and which will not.

To support these cases, all new sessions SHOULD be initiated with the N-bit set when possible. (As previously noted, the N-BIT is not carried in some exception cases.) When the first change in an EXP field is seen by the compressor, it MUST send a new MPLS_FULL_HEADER packet. At this point it may choose to not set the N-bit, or it may choose to wait to see if EXP changes are frequent. If EXP changes are observed to be frequent, the compressor SHOULD NOT set the N-bit. Operation when the N-bit is not set is described in Section 3.5. A compressor MAY be configured to never set the N-bit.

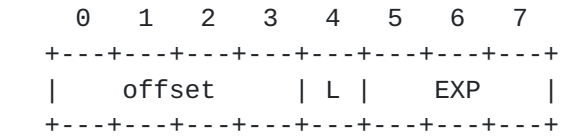
3.5. The EXP Compression Field

When the N-bit is not present, or is not set in the session's most recent associated MPLS_FULL_HEADER packet, the session is said to be in "EXP mode". In this mode one or more EXP fields are expected to change on as much as a per packet-by-packet basis. To support this mode a new field, called the EXP Compression field, is defined. When operating in EXP mode, at least one EXP Compression field is included in all compressed packet types. Compressed packet types are the types defined in [RFC2507], [RFC2508] and earlier in this document and begin with "COMPRESSED_."

Each EXP Compression field references a single MPLS label stack entry and provides the 3 bit EXP field to be used with that entry. Multiple EXP Compression fields may be included when more than stack entry is present. At most changes to 16 stack entries can be supported.

Only changes in EXP fields are carried, and the receiver MUST store the most recently received value. If no changes in EXP fields are observed, one EXP Compression field MUST be included in the compressed packet. In this case, the EXP Compression field SHOULD carry the EXP bits from the topmost stack entry. Per [LABELS], the top of the label stack appears earliest in the packet, and the bottom appears latest.

The format of the EXP Compression field is:



Offset indicates the count from the top of the stack to the corresponding label stack entry. To reference a particular stack entry, offset is incremented for each stack entry between the link layer header and the desired stack entry, e.g., zero indicates the top of the stack and two would indicate the third stack entry.

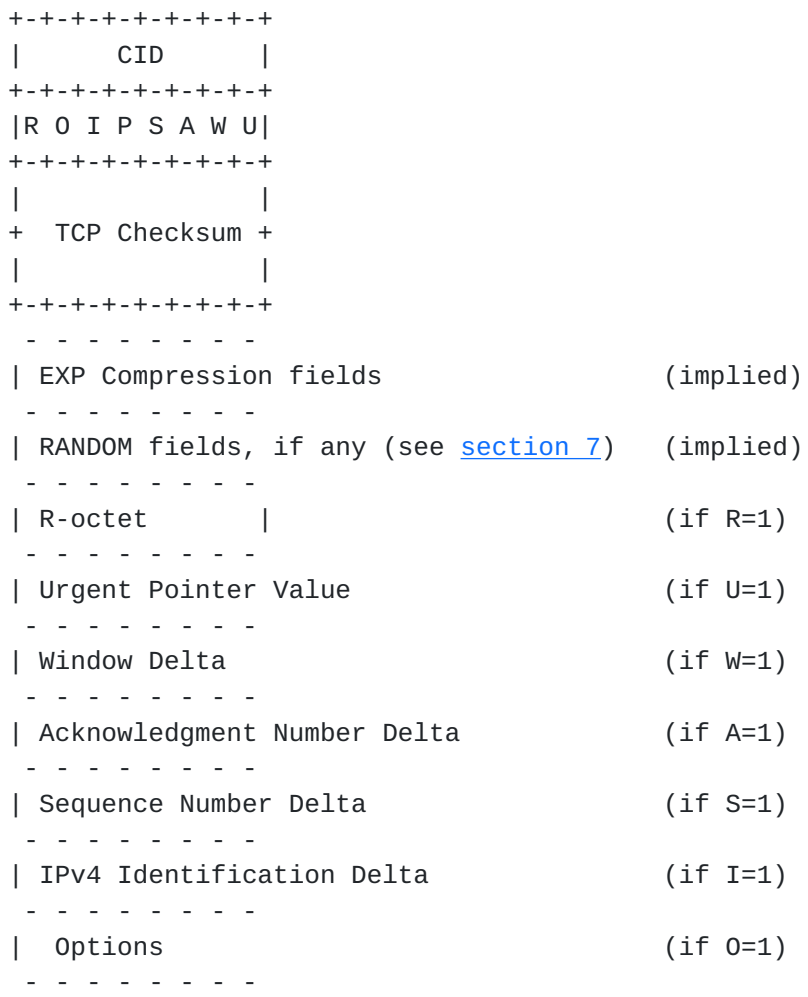
The L-bit is set in the last EXP Compression field present in the packet. A zero indicates that there is another EXP Compression field immediately following this field.

EXP is copied from the corresponding label stack entry.

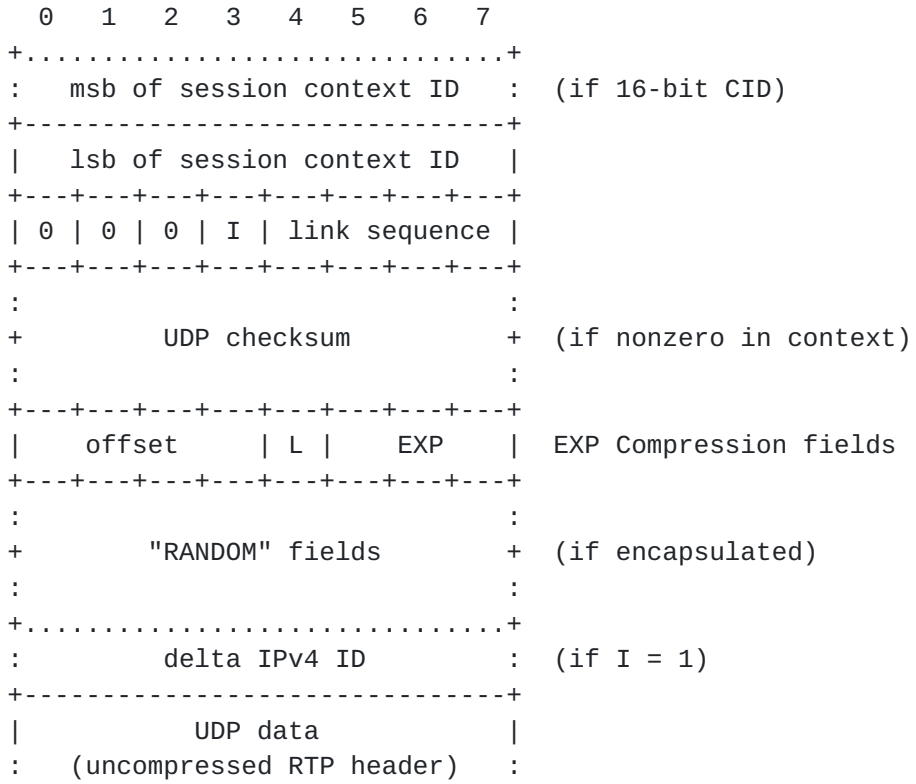
EXP Compression fields are carried as indicated in Section 3.2 and prior to the "RANDOM" fields defined in [RFC2507] and [RFC2508] in

other compressed packets. Two example modified packet formats are:

COMPRESSED_TCP format [RFC2507] for MPLS EXP mode session



COMPRESSED_UDP packet format [RFC2508] for MPLS EXP mode session



Note that there may be more than one EXP Compression field per packet.

3.6. Compatibility

MPLS and standard IP and IP/UDP/RTP compression can be performed on the same link. MPLS and IP sessions are distinguished via the use of MPLS_FULL_HEADER or FULL_HEADER packets. Once a session context is initialized, the CID implies whether MPLS headers are compressed. It is important to note that there is no separate MPLS session CID space. MPLS sessions share the TCP and non-TCP CID spaces. MPLS/IP/TCP sessions share the TCP CID space. Other sessions, including MPLS/IP/UDP/RTP sessions, share the non-TCP space.

It is also necessary for a compressor to know when a decompressor supports MPLS/IP compression. This information can be configured or negotiated. The default behavior is to assume that the decompressor does not support MPLS/IP compression.

Finally, the compressor needs to know the maximum number of MPLS stack entries the decompressor can process. This too can be configured or negotiated. The previously mentioned default is to

assume only one entry, i.e., one MPLS header, is supported.

4. Negotiating MPLS/IP Compression

The use of MPLS/IP compression over a particular link is a function of the link-layer protocol. As in [RFC2508], it is expected that such negotiation will be defined separately for PPP, for example. The following additional items SHOULD be negotiated:

- If MPLS/IP compression is supported
- The maximum stack depth supported

5. Security Considerations

No new security issues are raised by this document. Please see [RFC2507] and [RFC2508] for a detailed discussion of existing considerations associated with header compression.

6. References

[DIFFSERV] Faucheur, Wu, Davie, Davari, Vaananen, Krishnan, Cheval, "MPLS Support of Differentiated Services", draft-ietf-mpls-diff-ext-02.txt, October, 1999.

[LABELS] Rosen, Rekhter, Tappan, Farinacci, Fedorkow, Li, Conta, "MPLS Label Stack Encoding", draft-ietf-mpls-label-encaps-07.txt, September 1999.

[RFC1144] Jacobson, V., "TCP/IP Compression for Low-Speed Serial Links", RFC 1144, February 1990.

[RFC2507] Degermark, M., Nordgren, B. and S. Pink, "IP Header Compression", RFC 2507, February 1999.

[RFC2508] Casner, S., Jacobson, V., "Compressing IP/UDP/RTP Headers for Low-Speed Serial Link", RFC 2508, February 1999

[STD51] Simpson, W., "The Point-to-Point Protocol (PPP)", STD 51, RFC 1661, July 1994.

7. Authors' Address

Lou Berger
LabN Consulting, LLC
Voice: +1 301 468 9228
Email: lberger@labn.net

Jason Jeffords
Integral Access Inc.
321 Billerica Rd.
Chelmsford, MA 01824
Voice: +1 978 256 8833
Email: jjeffords@integralaccess.com