MPLS Working Group
INTERNET-DRAFT
Expiration Date: August 2000

Philip Matthews
Nortel Networks
February 2000

LDP/CR-LDP Session Reestablishment -- I'll Be Back
<draft-matthews-mpls-ldp-ibb-00.txt>

Status of this Memo

Abstract

This contribution proposes modifications to the LDP and CR-LDP protocols that allow an LDP or CR-LDP session to be reestablished using a new TCP connection if the old TCP connection goes down unexpectedly. It also proposes that, in certain situations, an LSR continue to use the label bindings associated with a session for a short time after the session goes down, to allow forwarding to continue uninterrupted while the two peer LSRs attempt to reestablish the session. These modifications allow an LSR to easily implement hitless software upgrades and hitless activity switches.

Conventions used in this document

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.

1. Introduction

Many recent router architectures decouple the control plane from the data plane, so that packet forwarding can continue even if the control software gets interrupted. One source of interruptions occurs during control switches; for example, when a router switches to a new version of the control software, or switches to a backup control processor in a control redundant system. It is possible to design a router to make these interruptions very brief, however, the nature of the TCP protocol is such that it is difficult to keep a TCP connection up across a control switch.

The current specification of the LDP and CR-LDP protocols ([LDP] and [CR-LDP]) state that if the TCP connection associated with an LDP or CR-LDP session goes down, then the session itself is terminated and all label bindings are discarded. For that reason, it is difficult today to build an LSR which can keep its LDP and CR-LDP sessions up across a control switch.

This contribution proposes modifications to the LDP and CR-LDP protocols that allow an LDP or CR-LDP session to be reestablished using a new TCP connection if the old TCP connection goes down. It also proposes that, in certain situations, an LSR continue to use the label bindings associated with a session for a short time after the session goes down, to allow forwarding to continue uninterrupted while the two peer LSRs attempt to reestablish the session. These changes allow a router to undergo a control switch with minimal disruption to the surrounding network.

This contribution proposes that the two peer LSRs negotiate at session establishment time whether they wish to allow the session to be restarted or not. If this capability is not agreed to, then the session operates as specified in [LDP] and [CR-LDP], and the new procedures described here are not used. The negotiation procedure is such that an LSR which implements these modifications can establish a session with a peer without any a priori knowledge of whether the peer supports these new procedures or not.

2. Overview of the Method

Say X and Y are two peer LSRs. When X and Y first establish an LDP or CR-LDP session, they include a new TLV, the Session Reestablishment Capability TLV, in the Initialization messages they exchange to negotiate the use of the procedures described in this draft.

Once Session Reestablishment Capability has been negotiated, the two peers use the message id field present in all LDP and CR-LDP

messages to track those messages that have been sent to their peer LSR but not yet processed. To enable them to do this, the two LSRs

treat the message id field as a 32-bit unsigned sequence number, incrementing it by one with each new message sent, and rolling it over to 0 after 2**31 - 1 is reached. This form of message id allocation is not required by the base LDP and CR-LDP specifications [LDP] and [CR-LDP], but is required by the procedures described in this draft.

Now say LSR X sends a message M to LSR Y. After it does so, X remembers that it has sent message M against the eventuality that TCP connection carrying M may be broken before Y receives the message.

When Y receives M, then it first processes the message according to the normal LDP or CR-LDP procedures. Y also records its new state in some manner that allows the state to be remembered across a session restart event. (For example, it may write the new state into non-volatile memory).

LSR Y then acks message M by using a new TLV, the Message Ack TLV, which contains the message id that X assigned to M. This Message Ack TLV is piggybacked on some message that Y happens to be sending back to X.

When X receives the ack, it knows that message M has been processed, so it can now discard the record it kept of M.

Now say some event happens that causes the TCP connection to drop. For example, Y might have control redundancy enabled and experience an activity switch. In this case, neither X or Y have any prior warning of the event. Alternatively, Y may be undergoing a software upgrade. In this case, Y may be able to shutdown the LDP session gracefully by sending a Notification message to X containing a new status code, the I'll-Be-Back status code, which indicates that Y hopes to reestablish the LDP or CR-LDP session shortly. In either case, Y is able to continue forwarding labelled packets without interruption (or with only a very brief interruption).

To reestablish the session, they first establish a new TCP connection, and then exchange Initialization messages. These Initialization messages contain a new TLV, the Want To Reestablish TLV, which indicates the willingness of each peer to reestablish the previous LDP or CR-LDP session. In the Initialization message sent by X, the Want To Reestablish TLV contains the message id of the last message that X managed to receive and process from Y before the old TCP connection went down. Similarly, the Initialization message from Y includes a Want To Reestablish TLV giving the message id of the last message that Y had received and processed from X.

Once Initialization messages have been successfully exchanged, the session has been reestablished. At this point, both peers know

precisely which messages were sent but not received, and can resend the missed messages. However, an LSR is not forced to send the missed messages in the precise way that they sent originally: it is free to send whatever messages it wishes to in whatever order it wishes to.

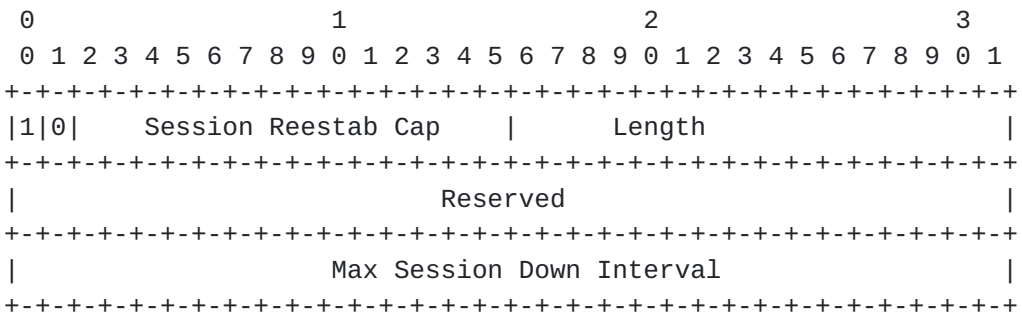
If the session is not reestablished, either because Y does not recover from the event, or because X and Y decide not to reestablish the session for some reason, then after a short interval X and Y both discard the label bindings associated with the session.

3. New TLVs and Status Codes

The following subsections describe the new TLVs introduced by the proposed method.

3.1 Session Reestablishment Capability TLV

The Session Restart Capability TLV can appear in the Initialization message to indicate willingness to follow the procedures described in this draft.



Reserved
This field must be set to zeros on transmission, and ignored on reception. (Future enhancements to this procedure might use this field for flags or other purposes).

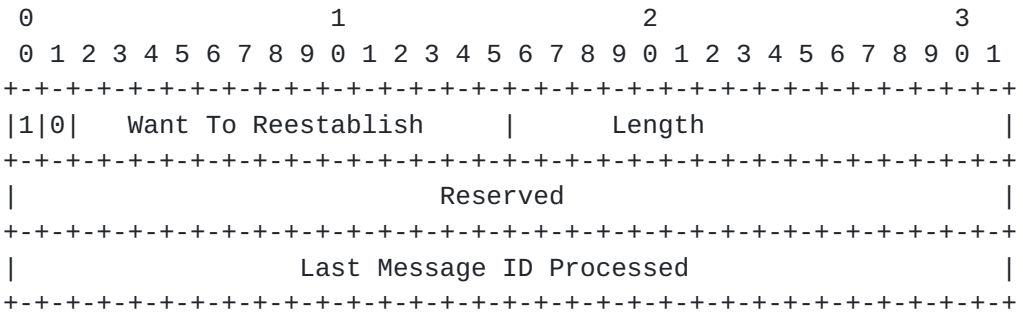
Max Session Down Interval
The maximum interval (in milliseconds) this LSR is willing to allow between the time it determines the TCP connection is broken and the time it determines the session has been successfully reestablished. Note that both ends propose Max Session Down Intervals -- the actual value is the minimum of the two proposed values.

The Session Reestablishment Capability TLV is an "optional" TLV according to the terminology of [LDP] and [CR-LDP]. It MUST appear only in the Initialization message and only when the LSR wishes to use the procedures described in this draft.

Because it is an optional TLV, the TLV has the U bit set to indicate that it should be ignored if it is not understood. This allows an LSR to propose the use of these procedures, but revert easily to standard [LDP] or [CR-LDP] operation if its peer does not understand the TLV. (See the procedures section below.)

3.2 Want To Reestablish TLV

The Want To Reestablish TLV can appear in the Initialization message to indicate willingness to reestablish a previous an [LDP] or [CR-LDP] session that had been prematurely terminated.



Reserved
This field must be set to zeros on transmission, and ignored on reception. (Future enhancements to this procedure might use this field for flags or other purposes).

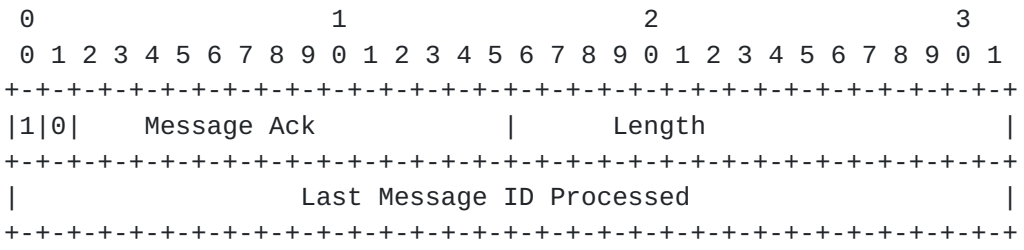
Last Message ID Processed
The ID of the last message which the sending LSR received and processed from the receiving LSR. The sending LSR may have received later messages from the receiving LSR, but the sending LSR did not complete processing of them and thus does not remember them.

The Want To Reestablish TLV is an "optional" TLV according to the terminology of [LDP] and [CR-LDP]. It MUST appear only in the Initialization message and only when the LSR wishes to restart a session using the procedures described in this draft.

Because it is an optional TLV, the TLV has the U bit set to indicate that it should be ignored if it is not understood. This allows an LSR to propose the restart of a session, but revert easily to standard [LDP] or [CR-LDP] operation if its peer does not understand the TLV. (See the procedures section below.)

3.3 Message Ack TLV

The Message Ack TLV can appear in any message to indicate acknowledgement of a message which the sending LSR has received from the receiving LSR.



Last Message ID Processed
The ID of the last message which the sending LSR received and processed from the receiving LSR.

Note that the ack is cumulative; that is, the use of this TLV acks not only the message specified but all previous messages. The receiving LSR MUST be able to accept gaps in the sequence of message IDs acked using this TLV. For example, it is acceptable for an LSR to include a Message Ack TLV with a value of 5, then not include any Message Ack TLV for a period of time, and then include a Message Ack TLV with a value of 12. This latter Message Ack TLV acks all messages from 6 to 12 inclusive.

The Message Ack TLV is an "optional" TLV according to the terminology of [LDP] and [CR-LDP]. It MAY appear in any message, but SHOULD appear only if the use of the procedures described in this draft has been agreed to by both peers.

Because it is an optional TLV, the TLV has the U bit set to indicate that it should be ignored if it is not understood. It also has the F bit cleared to indicate that it should not be forwarded to any other LSRs.

3.4 Status codes

This draft defines the following new status codes. See the procedures section for how they are used.

Status Code	E	Status Data
I'll Be Back	1	(tbd)
Session Rejected/ Parameters Max Session Down Interval	1	(tbd)
Session Rejected/	1	(tbd)

No Previous Session

Session Rejected/ Parameters Last Message ID Processed	1	(tbd)
Session Rejected/ Session Parameter Changed	1	(tbd)
Bad Message ID	1	(tbd)
Bad Message Ack	1	(tbd)
Out of Message IDs	1	(tbd)

4. New Procedures

4.1 Session Establishment

The procedures for session initialization are as specified in Section 2.5 of [LDP] with the following modifications.

a) An LSR which wishes to follow the procedures described in this draft includes a Session Reestablishment Capability TLV in the Initialization message it sends to its peer. An LSR which does not wish to follow the procedures described here does not include this TLV.

b) An LSR which receives an Initialization message containing a Session Reestablishment Capability TLV and which recognizes this TLV, but does not wish to follow the procedures described here ignores the TLV when processing the Initialization message. In particular, it SHOULD NOT send an "Unknown TLV" Status Code in reply.

c) An LSR which receives an Initialization message containing a Session Reestablishment Capability, and which wishes to follow the procedures described here, computes the minimum of the Max Session Down Interval specified in the message and its own Max Session Down Interval. If this value is acceptable, then it considers the TLV acceptable when processing the Initialization message, and it MUST use this computed value as the actual Max Session Down Interval for the duration of the session. If this value is not acceptable, then it SHOULD send an error Notification with a status code of "Session Rejected/Parameters Max Session Down Interval".

d) An LSR MUST both send a Session Reestablishment Capability TLV which is acceptable to its peer, and receive a Session Reestablishment Capability which is acceptable to it in order to use the procedures defined here for the remainder of the session. If it does not either send or receive a Session Reestablishment Capability TLV, then it SHOULD follow the procedures described in

[LDP] and [CR-LDP]. If both peers include Session Reestablishment

Capability TLVs in their Initialization messages, but the computed Max Session Down Interval is not acceptable to one or both peers, then the session is torn down as specified in [LDP].

4.2 Message IDs

The procedures for using Message IDs are as specified in [LDP] or [CR-LDP] with the following modifications.

a) Each LSR treats the Message ID field as an unsigned 32-bit sequence number.

b) An LSR MAY use any value it wishes for the Message ID of the Initialization message. The value it uses becomes the initial sequence number. Subsequent messages are sent with consecutive increasing sequence numbers, continuing with 0 after 2**32 - 1 is used.

c) When a session is reestablished, the old sequence of message IDs is broken and a new sequence is established with the message ID of the reestablishing Initialization message. For example, some implementations MAY elect to use the next number in the old sequence as the message ID of the Initialization message, while others MAY elect to restart the sequence at some fixed value.

d) An LSR which receives a message with a message ID that is not one greater than the message ID of the previous message (module 2**32), MUST terminate the session with a status code of "Bad Message ID".

e) An LSR MUST NOT reuse a Message ID until it has received an ack for its previous use. This ensures that the LSR can uniquely match message acks to messages. If an LSR is getting close to exhausting this interval, then it MAY elect to stop sending messages for a while to allow its peer a chance to ack some messages. Regardless of whether it pauses or not, an LSR must reserve the Message ID for a Notification message (with status code "Out of Message IDs") which it can use to terminate the session.

4.3 Message Acks

The procedures for processing received messages are as specified in [LDP] or [CR-LDP] with the following additions.

a) When processing a message, each LSR arranges to record in some way its new local state. Note that this does not require the LSR to remember the message or even remember the transition it underwent from its old local state to its new local state.

However, the processing SHOULD be done in a manner that is as atomic as possible, so that if a fault occurs during processing, the LSR restarts the session with the old state.

b) As part of the local state, each LSR keeps the message ID of the last message it processed.
c) Whenever an LSR sends a message to its peer, the LSR MAY elect to include a Message Ack TLV. The value of the Message Ack TLV SHOULD be the value of the last Message ID processed. In certain implementations, the routine filling in the Message Ack TLV may not learn of messages that have been newly processed for some time; in these implementations, the routine SHOULD use the most accurate value it knows. In all cases, an LSR MUST NOT ack a message that has not yet been processed.

d) An LSR MUST ack messages within a relatively short time after processing them.

e) The sequence of Message Ack values MUST be monotonically increasing (modulo 2**32). The value may repeat, but it may not go backwards, nor can it jump ahead to a message that has not been sent yet. If an LSR receives a Message Ack TLV which does not obey these rules, then it MUST terminate the session with a Notification message with a status code of "Bad Message Ack".

4.4 Session Termination

A sessions between peers who have negotiated the use of the Session Restart capability can be terminated in the following ways.

a) One or both peers can experience an event that causes the TCP connection to be terminated without warning. Events of this nature might include activity switches in a control redundant system.

b) One or both peers can terminate the session using a Notification message with a status code of "I'll Be Back".

c) One or both peers can terminate the session because their local TCP gave up, or because their local keepalive timer expired.

d) One or both peers can terminate the session using a Notification message with a status code OTHER than "I'll Be Back".

Sessions terminated in the fourth way SHOULD NOT restarted, and an LSR SHOULD reject any attempts to restart such sessions.

Sessions terminated in one of the first three ways are candidates for restarting. An LSR SHOULD continue to use the labels received

from its peer and honor the labels which it has distributed to its

peer until it determines that either the session has been restarted or it determines that the session cannot be successfully restarted. If an LSR determines that an session cannot be successfully restarted, it SHOULD discard any label bindings associated with the session.

An LSR determines that a session cannot be successfully restarted when one of the following occurs:

a) An interval longer than the computed max session down interval has elapsed since the LSR detected that the old TCP connection was broken.

b) A new session has been established, but the peers did not agree to make this session a continuation of the old session.

4.5 Session Reestablishment

The procedures for reestablishing a session are an modification of the procedures for establishing the session originally (as described in the section "Session Establishment" above).

a) The two LSR peers use the LDP Identifier and Receiver LDP Identifier fields of the Initialization message to uniquely identify the session being reestablished.

b) An LSR indicates its willingness to reestablish the previous session by including the Want To Reestablish TLV in its Initialization message.

c) A previous session can only be reestablished if both peers include the Want To Reestablish TLV in their Initialization messages, and each peer accept the value of the Want To Reestablish TLV that its receives.

d) If an LSR receives an Initialization message containing Want To Reestablish TLV, but it has no record of a previous session (perhaps because an interval greater than the computed max session down interval has elapsed since the previous session was terminated), then it rejects the Initialization message with a "Session Rejected/No Previous Session" status code.

e) If an LSR receives an Initialization message containing Want To Reestablish TLV, but it cannot reestablish the previous session at that point for some reason, then it rejects the Initialization message with a "Session Rejected/Parameters Last Message ID Processed" status code. (This could happen if the peer proposed a value which was out-of-range, or if, despite the peer proposing a

reasonable value, the local LSR simply cannot reestablish the session at that point, due to some internal restriction).

f) The reestablished session must have the same session parameters as the original session. Note that this does not mean that the Initialization messages used to reestablish the session must have exactly the same parameters as in the original exchange. Rather, it is the parameters that result from comparing the received Initialization message and the local configuration must be the same. A simple way to implement this is to send the computed session parameters from the original session in the reestablishing Initialization message.

g) If a peer detects that a session will be established with changed session parameters, then it SHOULD reject the session with a status code of "Session Rejected/Session Parameter Changed".

5. Security Considerations

There seems to be no difficulty in using these procedures with LDP or CR-LDP sessions that are protected using the MD5 signature option.

6. Areas for Further Study

This section discusses some possible areas for further study.

a) It might be useful to allow the session to be reestablished with new value for one or more session parameters. This would serve two purposes: one, it would provide a simple way to renegotiate session parameters, and two, it would provide a simple way of taking advantage of the new capabilities of upgraded control software. The main question to be answered here is: which session parameter changes can be reasonable supported? It is easy to see how a change in the KeepAlive interval can be accommodated, but what about changes to the label advertisement discipline or a decrease in the ATM label range?

b) It might also be useful to formalize methods of changing the transport addresses associated with the session. This would be particularly useful in control redundancy situations where the primary and backup LDP/CR-LDP entities have different IP addresses.

c) If the LSR which causes the TCP connection to drop plays the passive role in restarting the new session, then it must wait until its peer LSR initiates the session restart. If the underlying cause was an activity switch on the passive LSR, then the active LSR will not notice a problem until either the

KeepAlive timer expires or the local TCP times out. This may take

a while. It would be nice if the passive LSR could somehow kick the active LSR into action sooner. Unfortunately, there are security implications in providing such a mechanism. One solution might be to add an "I've Come Back" flag to the Hello message and then extend MD5 protection to these messages.

7. Acknowledgements

The original inspiration for this draft was the proposal by David Ward and John Scudder for restarting BGP sessions [WARD]. I have borrowed some of their terms, but the nature of LDP and CR-LDP (specifically DoD mode) forced me to adopt a different approach.

Thanks also to Peter Ashwood-Smith for helpful comments when I was working out the technical details behind this proposal.

8. References

[CR-LDP] Constraint-Based LSP Setup using LDP, draft-ietf-mpls-cr-ldp-04.txt

[LDP] LDP Specification, draft-ietf-mpls-ldp-05.txt

[WARD] BGP Notification Cease: I'll Be Back, draft-ward-bgp4-ibb-00.txt

9. Author's Address

Philip Matthews
Nortel Networks Corp.
P.O. Box 3511 Station C,
Ottawa, ON K1Y 4H7
Canada
Phone: +1 613-768-3262
philipma@nortelnetworks.com