Network Working Group
Internet Draft
Expiration Date: April 2002

Toby Smith (Laurel Networks)
Andrew G. Malis (Vivace Networks)
Jack Shaio (Vivace Networks)

Graceful Restart Mechanism for LDP

draft-smith-mpls-ldp-restart-00.txt

1. Status of this Memo

2. Abstract

This document proposes a lightweight mechanism that the LDP protocol may use to help minimize the impact introduced by transient interruptions to an LDP session's TCP connection, with a focus on connection preservation for signaled Layer 2 circuits. A new LDP Cork message request/response mechanism is specified. New message types are defined for the delivery of graceful restart events. Finally, procedures for utilizing this mechanism are detailed.

3. Introduction

The LDP protocol [1] provides label mapping information to its peer LSRs. In addition to providing label mappings for IP prefixes, LDP has recently been adopted as a signaling mechanism for the establishment of Layer 2 circuits between two provider edge LSRs [2, 3]. Customers expect these circuits, like the physical circuits they emulate, to be highly available connections.

Under some circumstances (planned outages, software upgrades), LDP may temporarily lose connectivity to its peer(s). In these circumstances, it is beneficial to the customer to maintain the LDP-established LSPs even in the (temporary) absence of an LDP session.

This draft describes a proposal for a lightweight mechanism which allows LDP LSRs to retain their forwarding state, even when the connection to the peer LSR is temporarily lost.

The procedure described in this draft has excellent scaling properties: the LDP state is preserved incrementally, such that after an unexpected restart of an LDP session, only the LDP activity not already acknowledged during the previous session needs to be resignaled. In the case of provisioned Layer 2 circuits, it is probable that no resignaling will be necessary.

The procedure described in this draft is minimally invasive to the LDP state machine and requires no changes to the LDP message processing procedures.

This mechanism may be used in conjunction with a mechanism for the preservation of IP forwarding state; when LDP is being used solely as a signaling mechanism for the establishment of Layer 2 transports, however, such coordination is not required.

The remainder of this document is organized as follows: A new LDP Cork message request/response mechanism is specified. New message types are defined for the delivery of graceful restart events. Finally, procedures for utilizing this mechanism are detailed.

4. Overview of Graceful Restart Mechanism

LDP LSRs which support this graceful restart mechanism signal this capability with an additional Graceful Restart TLV sent as part of the session's Initialization messages.

During normal session operation, each peer periodically issues a Cork message, defined below, which checkpoints the current label advertisement state between the peers. Each cork message is acknowledged by the far end.

If an LDP peer is able to recognize that it needs to temporarily drop its connection to its peer, this LSR (termed the Originating Peer) will send a special, final Cork message to each of its peer LSRs (termed the Receiving Peer(s)).

When the Receiving Peer receives a final Cork message, it responds with a corresponding final Cork message to the Originating Peer. Upon receiving the final Cork message response from each Receiving Peer, the Originating Peer may sever its TCP connection(s). All forwarding state corresponding to the cached state of the LDP protocol is preserved over the loss of connectivity with the LDP peer.

Once the Originating Peer's LDP state is able to be re-established, it reconnects to each of its Receiving Peers, following the standard procedures for establishing TCP connections as specified in [1].

When the TCP session to the Receiving Peer(s) has been re-established, the LSRs exchange Graceful Restart TLVs as part of their Initialization messages. This TLV contains that checkpoint information corresponding to the last exchanged Cork messages, which allows the LSRs to resume operation without readvertising any checkpointed label mapping information.

The details of the steps outlined in this section may be found in the Procedures section, below.

5. Message Formats

This section describes the new LDP message and TLV formats used by this document.

5.1 Cork Message

The LDP Cork message is sent periodically by each participating LSR. The Cork message may be used to checkpoint currently sent information, to acknowledge the reception of a previously received Cork message, or both.

The rate at which periodic Cork messages are sent is locally determined by each participating LSR, and is implementation dependent. For example, cork messages may be sent at regular intervals, or after a threshold of sent LDP messages has been exceeded. Cork updates are not necessary if the state of the LSR has not changed since the time the last Cork message was sent.

Cork messages with the Final Bit set are used to flush all currently pending label mapping and nexthop messages to the peer LSR, in anticipation of dropping the connection to the peer.

The encoding for the Cork Message is:

```
 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|       Cork (0x3F00)         |      Message Length           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Message ID                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Acknowledged Message ID                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|F|C|A|        Reserved         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Message ID

32-bit value used to identify this message.

Acknowledged Message ID

A 32-bit value used to acknowledge the reception of a prior Cork message from the sender. The receiver replies with a Cork message of its own, with this field set to the Message ID of the Cork message it is acknowledging. If the Acknowledgement Bit is not set (see below), this field MUST be ignored.

Final Bit

A single bit denoting whether this message is the final checkpointing Cork message that the receiver should expect to receive from the sender.

Checkpoint Bit

A single bit denoting that this Cork message is being used by the sender to checkpoint its currently sent label and address information. An LSR which receives a Cork message with the Checkpoint Bit set MUST acknowledge the reception of this message with a corresponding Cork message with the Acknowledgement Bit set (see below). Cork messages with the Checkpoint Bit set MUST contain a non-zero Message ID.

Acknowledgement Bit

A single bit denoting that this Cork message is being used by the sender to acknowledge the reception of a previously received Cork message. When the Acknowledgement Bit is set, the Acknowledged Message ID field MUST be set to the Message ID of the Cork message being acknowledged.

A single Cork message may have both the Checkpoint and Acknowledgement Bits set, allowing a single message to both checkpoint recently sent information, as well as acknowledge recently received Cork messages.

Reserved

These 13 bits MUST be filled with zeroes.

5.2 Graceful Restart TLV

The Graceful Restart TLV is contained within both the Originating and Receiving Peers' Initialization messages to denote their participation in the graceful restart protocol.

The encoding for the Graceful Restart TLV is:

```
 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| Graceful Restart (0x3F00) |           Length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Acknowledged Message ID                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Restart Timeout                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Acknowledged Message ID

If the LSR is establishing a connection to a peer for the first time, this field MUST be set to zero.

If an LSR is re-establishing a session with a remote peer with which it had previously exchanged Cork messages, and if the local LSR's Restart Timeout time has not expired, this value MUST contain the Message ID of the last successfully acknowledged Cork message received from the remote peer. If the Restart Timeout time has expired, this value MUST be reset to zero.

Restart Timeout

32-bit unsigned non-zero integer that indicates the number of seconds that the sending LSR is willing to wait for re-establishment of the TCP connection between the peers after a restart has begun. This timer is started when the current TCP connection is terminated. The Restart Timeout MUST be calculated by using the smaller of the values sent in the Graceful Restart TLV to the peer LSR and the Restart Timeout value in the Graceful Restart TLV received from the peer LSR.

6. Procedures

This section describes in detail the procedures which must be implemented by participating LSRs.

An LSR which is capable of participating in this mechanism includes a Graceful Restart TLV in the Initialization message it sends to its remote peer.

If the Initialization message received from the remote peer does not contain a Graceful Restart TLV, or if the value contained in the Acknowledged Message ID field is not the value expected from that peer, then the graceful restart mechanism MUST NOT be employed, and no Cork messages may be sent to the remote peer. In this case, if the local LSR has cached any state from a prior session to this peer, that cached state MUST be immediately discarded.

For two LSRs which have successfully exchanged Graceful Restart TLVs, the Restart Timeout value used by both LSRs is calculated to be the lesser of the values exchanged by the peers.

If this is the first time that the two LSRs have peered, or if the Restart Timeout time from a previous session has expired, the peering LSRs MUST include a value of zero in the Acknowledged Message ID field.

When the exchanged Acknowledged Message ID values are non-zero, and neither LSR's Restart Timeout time has expired, both peers MUST resume operation of the LDP session as if all checkpointed sent and received information is still active. Upon returning to such a state, the first message sent by each LSR to its peer MUST be a Cork message with the Acknowledgement Bit set, and the Acknowledged Message ID set to the value contained in the LSR's Graceful Restart TLV Acknowledged Message ID field. If the LSR is unable to restore its state for any reason, it MUST immediately send a Cork message with the Acknowledgement Bit set and containing an Acknowledged Message ID value of zero. In either case, after exchanging Initialization messages with non-zero Acknowledged Message ID values, the first messages exchanged between the peers MUST be Cork messages.

If an LSR which is re-establishing cached state after a restart receives an initial Cork message which does not match the value contained in the peer's Graceful Restart TLV, the receiving LSR MUST immediately discard any cached state, as the graceful restart has failed on the peer LSR.

After successfully negotiating the use of the graceful restart mechanism, and restoring cached state (if recovering from a prior restart), the peering LSRs resume normal LDP operation. Each LSR periodically checkpoints the label mapping and nexthop information that it has sent to its peer and issues an unsolicited Cork message with the Checkpoint Bit set to its peer. The sending LSR MUST NOT cache the current state of the sent session information until the remote peer acknowledges the receipt of the current Cork message.

If the local LSR knows a priori that it is about to restart, it may issue a Cork message with the Final Bit set. After sending a Cork message with the Final bit set, the sending LSR MUST NOT send any further Label Mapping, Label Withdraw, Address, or Address Withdraw messages to the receiving peer.

An LSR which receives a Cork message from its peer with the Checkpoint Bit set MUST acknowledge the receipt of this message by responding to the sending peer with a Cork message with the Acknowledgement Bit set. The receiving LSR MUST cache all received session information from the remote peer before acknowledging the reception of a Checkpoint Cork message.

If the received Cork message's Final bit is set, the receiving peer immediately sends any pending Label Mapping, Label Withdraw, Address, and Address Withdraw messages to the sending peer, followed by a Cork message with the Final bit set in response. This Cork message may also serve to acknowledge receipt of the sending peer's Final Cork message. After sending the Cork message, the receiving peer MUST not send any more Label Mapping, Label Withdraw, Address, or Address Withdraw messages to the sending peer.

An LSR which is expecting to be restarted initiates the graceful restart by sending a Cork message with the Final bit set to its peer. This LSR may restart upon receiving both a corresponding Final Cork message from its peer, and upon receiving a Acknowledgement Cork message from its peer. These two messages may be consolidated into a single message with the Final, Checkpoint and Acknowledgement Bits set.

LSRs participating in this graceful restart mechanism do not expect to see a fatal Notification message from their remote peer before restarting. If an LSR sends a fatal Notification message to its remote peer, or receives a fatal Notification from its remote peer, the LSR MUST discard any cached LDP state immediately.

7. Operational Considerations

This document describes a mechanism for the graceful re-establishment of LDP sessions, with a focus on providing a simple signaling recovery mechanism for Layer 2 transport LSPs. Given that the establishment of IP LSPs via LDP relies upon the existence of an underlying IGP to determine the network topology, a complete graceful restart mechanism requires a degree of coordination between LDP and its underlying IGP when restarting. This document does not address ways in which the IGP state may be preserved during a graceful restart.

8. Security Considerations

Given that this document describes a mechanism for preserving LDP session state during periods of lost connectivity, there may be concern that this proposal introduces new security risks. However, since the re-establishment of the LDP session is based upon the same mechanisms described in [1], and since the cached LDP session state is only eligible for use if an LDP session is re-established to a peer which had previously been peering with the LSR, the authors believe that this proposal does not impact the underlying security model of LDP.

9. References

[1] "LDP Specification", L. Andersson, P. Doolan, N. Feldman, A. Fredette, B. Thomas. RFC3036

[2] "Transport of Layer 2 Frames Over MPLS", draft-martini-l2circuit-trans-mpls-08.txt. (work in progress)

[3] "MPLS-based Layer 2 VPNs", Kompella, et. al., draft-kompella-mpls-l2vpn-02.txt. (work in progress)

10. Author Information

Toby Smith
Laurel Networks, Inc.
1300 Omega Drive
Pittsburgh, PA 15205
Email: tob@laurelnetworks.com

Andrew G. Malis
Vivace Networks, Inc.
2730 Orchard Parkway
San Jose, CA 95134
Phone: +1 408 383 7223
Email: Andy.Malis@vivacenetworks.com

Jack Shaio
Vivace Networks, Inc.
2730 Orchard Parkway
San Jose, CA 95134
Phone: +1 408 432 7623
Email: Jack.Shaio@vivacenetworks.com