Network Working Group
Internet Draft
Category: Informational
Expiration Date: March 2003
September 2002

Adrian Farrel
Movaz Networks

Applicability Statement for Restart Mechanisms for the Label Distribution Protocol

draft-farrel-mpls-ldp-restart-applic-00.txt

Status of this Memo

Abstract

Multiprotocol Label Switching (MPLS) systems will be used in core networks where system downtime must be kept to a minimum. Similarly, where MPLS is at the network edges (for example, in Provider Edge routers) system downtime must also be kept as small as possible. Many MPLS Label Switching Routers (LSRs) may, therefore, exploit Fault Tolerant (FT) hardware or software to provide high availability of the core networks.

The details of how FT is achieved for the various components of an FT LSR, including the switching hardware and the TCP stack are implementation specific. How the software module itself chooses to implement FT for the state created by the Label Distribution Protocol (LDP) is also implementation specific but there are several issues in the LDP specification in RFC 3036 "LDP Specification" that make it difficult to implement an FT LSR using the LDP protocols without some extensions to those protocols.

Proposals have been made in "Fault Tolerance for the Label Distribution Protocol (LDP)" [LDP-FT] and "Graceful Restart Mechanism

for LDP" [LDP-RESTART] to address these issues.

This document gives guidance on when it is advisable to implement some form of LDP restart mechanism and which approach might be more suitable. The issues and extensions described here are equally applicable to RFC 3212, "Constraint-Based LSP Setup Using LDP" (CR-LDP).

1. Requirements of an LDP FT System

MPLS is a technology that will be used in core networks where system downtime must be kept to an absolute minimum. Similarly, where MPLS is at the network edges (for example, in PE routers in RFC2547) system downtime must also be kept as small as possible.

Many MPLS LSRs may, therefore, exploit FT hardware or software to provide high availability (HA) of core networks.

In order to provide HA, an MPLS system needs to be able to survive a variety of faults with minimal disruption to the Data Plane, including the following fault types:

- failure/hot-swap of the switching fabric in an LSR

- failure/hot-swap of a physical connection between LSRs

- failure of the TCP or LDP stack in an LSR

- software upgrade to the TCP or LDP stacks in an LSR.

The first two examples of faults listed above may be confined to the Data Plane in which case such faults can be handled by providing redundancy in the Data Plane which is transparent to LDP operating in the Control Plane. However, the failure of the switching fabric or a physical link may have repercussions in the Control Plane since signaling may be disrupted.

The third example may be caused by a variety of events including processor or other hardware failure, and software failure.

Any of the last three examples may impact the Control Plane and will require action in the Control Plane to recover. Such action should be designed to avoid disrupting traffic in the Data Plane. This is possible because many recent router architectures separate the Control and Data Planes such that forwarding can continue unaffected by recovery action in the Control Plane.

In other scenarios, the Data and Control Planes may be impacted by a fault but the needs of HA require the coordinated recovery of the Data and Control Planes to state that existed before the fault.

The provision of protection paths for MPLS LSP and the protection of links, IP routes or tunnels through the use of protection LSPs is outside the scope of this document. See [MPLS-RECOV] for further information on this subject.

2. General Considerations

In order that the Data and Control Plane states may be successfully recovered after a fault, procedures are required to ensure that the state held on a pair of LDP peers (at least one of which was affected directly by the fault) are synchronized. Such procedures must be implemented in the Control Plane software modules on the peers using Control Plane protocols.

The required actions may be operate fully after the failure (reactive recovery) or may contain elements that operate before the fault in order to minimize the actions taken after the fault (proactive recovery). It is rarely feasible to implement actions that operate solely in advance of the failure and do not require any further processing after the failure (preventive recovery) - this is because of the dynamic nature of signaling protocols and the unpredictability of fault timing.

Reactive recovery actions may include full re-signaling of state, re-synchronization of state between peers and synchronization based on checkpointing.

Proactive recovery actions may include hand-shaking state transitions and checkpointing.

3. Specific Issues with the LDP Protocol

LDP uses TCP to provide reliable connections between LSRs over which to exchange protocol messages to distribute labels and to set up LSPs. A pair of LSRs that have such a connection are referred to as LDP peers.

TCP enables LDP to assume reliable transfer of protocol messages. This means that some of the messages do not need to be acknowledged (for example, Label Release).

LDP is defined such that if the TCP connection fails, the LSR should immediately tear down the LSPs associated with the session between the LDP peers, and release any labels and resources assigned to those LSPs.

It is notoriously hard to provide a Fault Tolerant implementation of TCP. To do so might involve making copies of all data sent and received. This is an issue familiar to implementers of other TCP applications such as BGP.

During failover affecting the TCP or LDP stacks, therefore, the TCP connection may be lost. Recovery from this position is made worse by the fact that LDP control messages may have been lost during the connection failure. Since these messages are unconfirmed, it is possible that LSP or label state information will be lost.

The solution to this problem must at the very least include a change to the basic requirements of LDP so that the failure of an LDP session does not require that associated LDP or forwarding state be torn down.

Any changes made to LDP in support of recovery processing must meet the following requirements:

- offer backward-compatibility with LSRs that do not implement the extensions to LDP

- preserve existing protocol rules described in [RFC3036] for handling unexpected duplicate messages and for processing unexpected messages referring to unknown LSPs/labels.

Ideally, any solution applicable to LDP should be equally applicable to CR-LDP.

4. Summary of the Features of LDP FT

LDP Fault Tolerance extensions are described in [LDP-FT]. This approach involves:

- negotiation between LDP peers of the intent to support extensions to LDP that facilitate recovery from failover without loss of LSPs

- selection of FT survival on a per LSP/label basis or for all labels on a session

- sequence numbering of LDP messages to facilitate acknowledgement and checkpointing

- acknowledgement of LDP messages to ensure that a full handshake is performed on those messages either frequently (such as per message) or less frequently as in checkpointing

- solicitation of up-to-date acknowledgement (checkpointing) of previous LDP messages to ensure the current state is secured, with an additional option that allows an LDP partner to request that state is flushed in both directions if graceful shutdown is required

- a timer to control for how long LDP and forwarding state should be retained after LDP session failure before being discarded if LDP communications are not re-established

- exchange of checkpointing information on LDP session recovery to establish what state has been retained by recovering LDP peers

- re-issuing lost messages after failover to ensure that LSP/label state is correctly recovered after reconnection of the LDP session.

The FT procedures in [LDP-FT] concentrate on the preservation of label state for labels exchanged between a pair of adjacent LSRs when the TCP connection between those LSRs is lost. There is no intention within these procedures to support end-to-end protection for LSPs.

5. Summary of the Features of LDP Graceful Restart

LDP graceful restart extensions are defined in [LDP-RESTART]. This approach involves:

- negotiation between LDP peers of the intent to support extensions to LDP that facilitate recovery from failover without loss of LSPs

- a mechanism whereby an LSR that restarts can relearn LDP state by resynchronization with its peers

- use of the same mechanism to allow LSRs recovering from an LDP session failure to resynchronize LDP state with their peers provided that at least one of the LSRs has retained state across the failure or has itself resynchronized state with its peers

- a timer to control for how long LDP and forwarding state should be retained after LDP session failure before being discarded if LDP communications are not re-established

- a timer to control the length of the period during which resynchronization of state between adjacent peers should be completed

The procedures in [LDP-RESTART] are applicable to all LSRs, both those with the ability to preserve forwarding state during LDP restart and those without. An LSRs that can not preserve its MPLS forwarding state across the LDP restart would impact MPLS traffic during restart, but by implementing a subset of the mechanisms in [LDP-RESTART] it can minimize the impact if their neighbor(s) are capable of preserving their forwarding state across the restart of their LDP sessions or control planes by implementing the mechanism in [LDP-RESTART].

6. Applicability Considerations

This section considers the applicability of fault tolerance schemes within LDP networks and considers issues that might lead to the choice of one method or another. Many of the points raised below should be viewed as implementation issues rather than specific drawbacks of either solution.

6.1 General Applicability

The procedures described in [LDP-FT] and [LDP-RESTART] are intended to cover two distinct scenarios. In Session Failure the LDP peers at the ends of a session remain active, but the session fails and is restarted. In Node Failure the session fails because one of the peers has been restarted (or at least, the LDP component of the node has been restarted). These two scenarios have different implications for the ease of retention of LDP state within an individual LSR, and are described in sections below.

These techniques are only applicable in LDP networks where at least one LSR has the capability to retain LDP signaling state and the associated forwarding state across LDP session failure and recovery. In [LDP-RESTART] the LSRs retaining state do not need to be adjacent to the failed LSR or session.

If traffic is not to be impacted, both LSRs at the ends of an LDP session must at least preserve forwarding state. Preserving LDP state is not a requirement to preserve traffic.

[LDP-FT] requires that the LSRs at both ends of the session implement the procedures that is describes. Thus, either traffic is preserved and recovery resynchronizes state, or no traffic is preserved and the LSP fails.

Further, to use the procedures of [LDP-FT] to recover state on a session both LSRs must have a

[LDP-RESTART] is scoped to support preservation of traffic if both LSRs implement the procedures that it describes. Additionally, it functions if only one LSR on the failed session supports retention of forwarding state, and implements the mechanisms in the document - in this case traffic will be impacted by the session failure, but the forwarding state will be recovered on session recovery. Further, in the event of simultaneous failures, [LDP-RESTART] is capable of relearning and redistributing state across multiple LSRs by combining its mechanisms with the usual LDP message exchanges of [RFC 3036].

6.2 Session Failure

In Session Failure an LDP session between two peers fails and is restarted. There is no restart of the LSRs at either end of the session and LDP continues to function on those nodes.

In these cases, it is simple for LDP implementations to retain LDP state associated with the failed session and to associate the state with the new session when it is established. Housekeeping may be applied to determine that the failed session is not returning and to release the old LDP state. Both [LDP-FT] and [LDP-RESTART] handle this case.

6.3 Controlled Session Failure

In some circumstances the LSRs may know in advance that an LDP session is going fail - perhaps a link is going to be taken out of service.

[RFC 3036] includes provision for controlled shutdown of a session.

[LDP-FT] and [LDP-RESTART] allow resynchronization of LDP state upon re-establishment of the session.

[LDP-FT] offers the facility to both checkpoint all state before the shut-down, and to quiesce the session so that no new state changes are attempted between the checkpoint and the shut-down. This means that on recovery, resynchronization is simple and fast.

[LDP-RESTART] resynchronizes all state on recovery regardless of the nature of the shut-down.

6.4 Node Failure

Node Failure describes events where a whole node is restarted or where the component responsible for LDP signaling is restarted. Such an event will be perceived by the LSR's peers as session failure, but the restarting node sees the restart as full re-initialization.

The restarting LSR may have preserved state from before the restart. The ways to do this are numerous and implementation specific and it is not the purpose of this document to espouse one mechanism or another nor even to suggest how this might be done. If state has been preserved across the restart, synchronization with peers can be carried out as though recovering from Session Failure as in the previous section. Both [LDP-FT] and [LDP-RESTART] support this case.

It is also possible that the restarting LSR has not preserved any state. In this case [LDP-FT] is of no help. [LDP-RESTART] however allows the restarting LSR to relearn state from each adjacent peer through the processes for resynchronizing after Session Failure. Further, in the event of simultaneous failure of multiple adjacent nodes, the nodes at the edge of the failure zone can recover state from their active neighbors and distribute it to the other recovering LSRs without any failed LSR having to have saved state.

6.5 Controlled Node Failure

In some cases (hardware repair, software upgrade, etc.) node failure may be predictable. In these cases all sessions with peers may be shutdown and existing state retention may be enhanced by special actions.

[LDP-FT] checkpointing and quiesce may be applied to all sessions so that state is up-to-date.

As above, [LDP-RESTART] does not require that state is retained by the restarting node, but can utilize it if it is.

6.6 Speed of Recovery

Speed of recovery is impacted by the amount of signaling required.

If forwarding state is preserved on both LSRs on the failed session then the recovery time is constrained by the time to resynchronize the state between the two LSRs.

[LDP-FT] may resynchronize very quickly. In a stable network this resolves to a handshake of a checkpoint. At the most, resynchronization involves this handshake plus an exchange of messages to handle state changes since the checkpoint was taken. Implementations that support only the periodic checkpointing subset of [LDP-FT] are more likely to have additional state to resynchronize.

[LDP-RESTART] must resynchronize state for all label mappings that have been retained; this may require a two-way message exchange for each label in downstream on demand mode. At the same time, resources that have be retained by a restarting upstream LSR but are not actually required because they have been released by the downstream LSR (perhaps because it was in the process of releasing the state) must be held for the full resynchronization time to ensure that they are not needed.

The impact of recovery time will vary according to the use of the network. Both [LDP-FT] and [LDP-RESTART] allow advertisement of new labels while resynchronization is in progress. Issues to consider are re-availability of falsely retained resources and conflict between retained label mappings and newly advertised ones since this may cause incorrect forwarding of data.

6.7 Scalability

Scalability is largely the same issue as speed of recovery and is governed by the number of LSPs managed through the failed session(s).

Note that there are limits to how small the resynchronization time in [LDP-RESTART] may be made given the capabilities of the LSRs, the throughput on the link between them, and the number of labels that must be resynchronized.

Impact on normal operation should also be considered.

[LDP-FT] requires acknowledgement of all messages. These acknowledgements may be deferred as for checkpointing described in section 6.4, or may be frequent. Although acknowledgements can be piggy-backed on other state messages, an option for frequent acknowledgement is to send a message solely for the purpose of acknowledging a state change message. Such an implementation would clearly be unwise in a busy network.

[LDP-RESTART] has no impact on normal operations.

6.8 Rate of Change of LDP State

Some networks do not show a high degree of change over time, such as those using targeted LDP sessions; others change the LDP forwarding state frequently, perhaps reacting to changes in routing information on LDP discovery sessions.

Rate of change of LDP state exchanged over an LDP session depends on the application for which the LDP session is being used. LDP sessions used for exchanging <FEC, label> bindings for establishing hop by hop LSPs will typically exchange state reacting to IGP changes. Such exchanges could be frequent. On the other hand LDP sessions established for exchanging MPLS Layer 2 VPN FECs will typically exhibit a smaller rate of state exchange.

In [LDP-FT] two options exist. The first uses a frequent (up to per-message) acknowledgement system which is most likely to be applicable in a more dynamic system where it is desirable to preserve the maximum amount of state over a failure to reduce the level of resynchronization required and to speed the recovery time.

The second option in [LDP-FT] uses a less-frequent acknowledgement scheme known as checkpointing. This is particularly suitable to networks where changes are infrequent or bursty.

[LDP-RESTART] resynchronizes all state on recovery regardless of the rate of change of the network before the failure. This consideration is thus not relevant to the choice of [LDP-RESTART].

6.9 Implementation Complexity

Implementation complexity has consequences for the implementer and also for the deployer since complex software is more error prone and harder to manage.

[LDP-FT] is a more complex solution than [LDP-RESTART]. In particular, [LDP-RESTART] does not require any modification to the normal signaling and processing of LDP state changing messages.

6.10 Implementation Robustness

In addition to the implication for robustness associated with complexity of the solutions, consideration should be given to the effects of state preservation on robustness.

If state has become incorrect for whatever reason then state preservation may retain incorrect state. In extreme cases it may be that the incorrect state is the cause of the failure in which case preserving that state would be bad.

When state is preserved, the precise amount that is retained is an implementation issue. The basic requirement is that forwarding state is retained (to preserve the data path) and that that state can be accessed by the LDP software component.

In both solutions, if the forwarding state is incorrect and is retained, it will continue to be incorrect. Both solutions have a mechanism to housekeep and free unwanted state after resynchronization is complete. [LDP-RESTART] may be better at eradicating incorrect forwarding state because it replays all messages exchanges that caused the state to be populated.

In [LDP-RESTART] no more data than the forwarding state needs to have been saved by the recovering node. All LDP state may be relearned by message exchanges with peers. Whether those exchanges may cause the same incorrect state to arise on the recovering node is an obvious concern.

In [LDP-FT] the forwarding state must be supplemented by a small amount of state specific to the protocol extensions. LDP state may be retained directly or reconstructed from the forwarding state. The same issues apply when reconstructing state but are mitigated by the fact that this is likely a different code path. Errors in the retained state specific to the protocol extensions will persist.

6.11 Interoperability and Backward Compatibility

It is important that new additions to LDP interoperate with existing implementations at least in provision of the existing levels of function.

Both [LDP-FT] and [LDP-RESTART] do this through rules for handling the absence of the FT optional negotiation object during session initialization.

Additionally, [LDP-RESTART] is able to perform limited recovery (that is, redistribution of state) even when only one of the participating LSRs supports the procedures. This may offer considerable advantages in interoperation with legacy implementations.

6.12 Interaction With Other Label Distribution Mechanisms

Many LDP LSRs also run other label distribution mechanisms. These include management interfaces for configuration of static label mappings, other distinct instances of LDP, and other label distribution protocols. The last example includes traffic engineering label distribution protocol that are used to construct tunnels through which LDP LSPs are established.

As with re-use of individual labels by LDP within a restarting LDP system, care must be taken to prevent labels that need to be retained by a restarting LDP session or protocol component from being used by another label distribution mechanism since that might compromise data security amongst other things.

It is a matter for implementations to avoid this issue through the use of techniques such as a common label management component or segmented label spaces.

6.13 Applicability to CR-LDP

Although CR-LDP [RFC 3212] is not a direct consideration of either [LDP-FT] or [LDP-RESTART], both are suitable for application to CR-LDP LSPs whether in a network entirely based on CR-LDP or in one that is mixed between LDP and CR-LDP.

7. Security Considerations

This document is informational and introduces no new security concerns.

The security considerations pertaining to the original LDP protocol [RFC3036] remain relevant.

[LDP-RESTART] introduces the possibility of additional denial-of-service attacks. All of these attacks may be countered by use of an authentication scheme between LDP peers, such as the MD5-based scheme outlined in [LDP].

In MPLS, a data mis-delivery security issue can arise if an LSR continues to use labels after expiration of the session that first caused them to be used. Both [LDP-FT] and [LDP-RESTART] are open to this issue.

8. Intellectual Property Considerations

Parts of [LDP-FT] are the subject of a patent application by Data Connection Ltd.

Parts of [LDP-RESTART] are the subject of patent applications by Juniper Networks and Redback Networks.

In all cases, the parties have indicated that if technology is adopted as a standard they agree to license, on reasonable and non-discriminatory terms, any patent rights they obtain covering such technology to the extent necessary to comply with the standard.

9. References

9.1 Normative References

[RFC2026] Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3036] Andersson, L., et. al., LDP Specification, RFC 3036, January 2001.

[LDP-FT] Farrel, A., et al., Fault Tolerance for the Label Distribution Protocol (LDP), draft-ietf-mpls-ldp-ft-06.txt, September 2002, work in progress.

[LDP-RESTART] Leelanivas, M., et al., Graceful Restart Mechanism for LDP, draft-ietf-ldp-restart-05.txt, September 2002, work in progress.

9.2 Informational References

[MPLS-RECOV] Sharma, Hellstrand, et al., Framework for MPLS-based Recovery, draft-ietf-mpls-recovery-frmwrk-07.txt, September 2002, work in progress.

[RFC3212] Jamoussi, B., et. al., Constraint-Based LSP Setup using LDP, RFC 3212, January 2002.

10. Acknowledgements

The author would like to thank the authors of [LDP-FT] and [LDP-RESTART] for their work on fault tolerance of LDP. Many thanks to Yakov Rekhter, Rahul Aggarwal and Manoj Leelanivas for their considered input to this applicability statement.

11. Author Information

Adrian Farrel
Movaz Networks, Inc.
7926 Jones Branch Drive, Suite 615
McLean, VA 22102
Phone: +1 703-847-1867
Email: afarrel@movaz.com

12. Full Copyright Statement