Fast rerouting mechanism for a protected Label Switched Path

Lemma Hundessa and Jordi Domingo Pascual
Departament d'Arquitectura de Computadors
Universitat Politècnica de Catalunya (UPC)
C/ Jordi Girona 1-3, 08034. Barcelona
{hundessa, jordid} @ac.upc.es

Abstract–**The paper presents a fast and efficient mechanism for rerouting traffic when there is a link failure or congestion problem in Multiprotocol Label Switching (MPLS) networks. Our method uses a predefined, alternative Label Switched Path (LSP) in order to restore traffic and significantly improves the performance in comparison to similar, existing proposals. We show that our proposal is not only able to reduce the average traffic delay due to path restoration, but also eliminates the problem of packet disorder. The latter - as is well known - can have significant impact on the overall, end-to-end performance.**
We have simulated our mechanism using the MPLS Network Simulator (MNS) and the performance metrics were compared to other proposals. The simulation results show that our new mechanism provides a significant improvement on average delay while eliminating packet disorder. The combination of these two improvements helps to minimize the effects of link failure and/or congestion. This facilitates the rapid release of network resources and improves the end-to-end performance of MPLS networks.

Keywords–**Fast rerouting, Label Switched Path (LSP), Switchover, Alternative LSP, Label Switching Router(LSR).**

I. INTRODUCTION

The emergence of Multi Protocol Label Switching (MPLS) as part of the Internet forwarding architecture is a significant contribution to traffic engineering [1][2][3]. Some of the elements that make up the MPLS traffic engineering solution are: Label Switched Paths (LSPs), appropriate path discovery in a network, traffic assignment to paths and rapid response to topology changes.

Given that network topologies are never stable over time, rapid response to link failures and/or congestion by means of rerouting is critical. There are two basic methods for LSP recovery: (1) Dynamic rerouting and (2) Fast rerouting [4]. The former works by establishing an alternative Label Switched Path or LSP segment on-demand, after the occurrence of a fault. Fast rerouting uses pre-established alternative LSPs or LSP segments. We focus on improving current mechanisms for fast rerouting.

The paper is organized as follows: In section 2, we discuss the strategies pursued for fast rerouting. Some of the related work for improving them is discussed in section 3. Our proposal is explained in section 4 and the detailed algorithm is described in section 5. In section 6, we explain the simulation methodology and present our results. In the final section, we summarize our conclusions and offer suggestions for future work.

This work has been partially supported by the Spanish ministry (CICYT) under contract number TEL99-1117-C03-03. Lemma Hundessa is granted by Spanish Agency for International Cooperation (A.E.C.I).

II. REROUTING STRATEGIES

As explained above, fast rerouting uses pre-established LSPs. When a fault is detected, the protected traffic is switched over to the alternative LSP(s). Setting pre-established alternative paths, results in a faster switchover compared to establishing new alternative paths on-demand [5][4][6][7]. However, fast rerouting may lead to the use of non-optimal alternative LSPs. Global optimization algorithms that can be computed at the ingress point of the LSP have been proposed to alleviate this drawback [7]. The combination of both fast rerouting and optimal path computation would be the best solution for service restoration. The optimal, alternative LSP discovery method is not within the scope of this paper.

Fast rerouting can be accomplished by protection mechanisms that are activated locally or that are global in scope. Local repair uses an alternative Label Switched Path (LSP) that serves as a bypass from the point of protection to next LSR node or to the destination. The techniques proposed for local repairs in MPLS networks are splicing and stacking [7]. Global repair is activated on an end-to-end basis. That is, an alternative LSP is pre-established from ingress to egress nodes of the path to be protected.

If local repair is attempted to protect an entire LSP, each intermediate Label Switching Router (LSR) must have the capability of initiating alternative, pre-established LSPs. This is because it would be impossible to predict where failure may occur within an LSP. A very high cost has to be paid in terms of complex computations and extensive signalling required to establish alternative LSPs from each intermediate LSR to the egress LSR. For this reason, we have chosen the combination of local and global repair strategy for our mechanism. Our approach is similiar to the one adopted in [5].

III. RELATED WORK

In [5], when a failure is detected, the traffic is sent backwards to the ingress LSR using a pre-established LSP - within the protected segment (we call it backwards LSP). When the ingress LSR receives the first packet from the backwards LSP, the traffic flow is redirected to the alternative LSP that was established previously between ingress and egress LSRs (Fig. 1).

In Fig. 1, the ingress and egress nodes respectively are LSR0 and LSR4. The protected LSP is formed by the LSR nodes 0,1,2,3 and 4. If a link failure is detected by LSR3 - as shown in the figure, the backward LSP will consist of the nodes 3,2,1

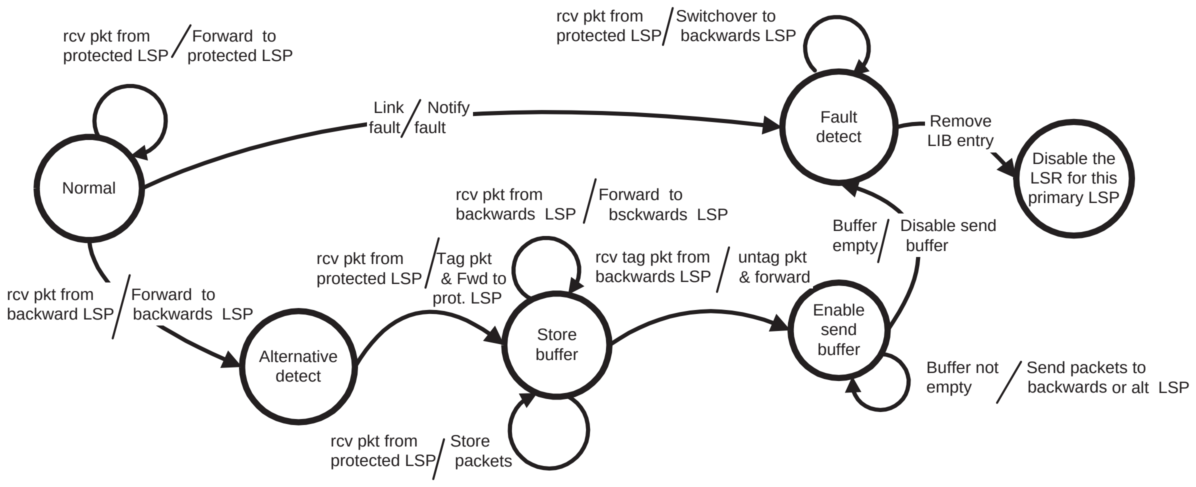
rcv pkt from protected LSP / Forward to protected LSP
Normal
Link fault / Notify fault
rcv pkt from protected LSP / Switchover to backwards LSP
Fault detect
Remove LIB entry
Disable the LSR for this primary LSP
rcv pkt from backwards LSP / Forward to bsckwards LSP
Buffer empty / Disable send buffer
rcv pkt from protected LSP / Tag pkt & Fwd to prot. LSP
rcv tag pkt from backwards LSP / untag pkt & forward
rcv pkt from backward LSP / Forward to backwards LSP
Alternative detect
Store buffer
Enable send buffer
Buffer not empty / Send packets to backwards or alt LSP
rcv pkt from protected LSP / Store packets

Fig. 2. State machine diagram.

and 0. What we name as the alternative LSP will be formed by the LSR nodes 0,5,6,7,8 and 4.

As soon as an LSR node belonging to the protected LSP detects a fault, a switchover is established and packets are sent back through the newly activated backwards LSP Fig. 1. The first packet that is sent back is used as a fault detect notification. Until the fault notification arrives at the ingress LSR, packets are sent via the already broken protected LSP. These packets will experience a two-way delay while traversing the backwards loop from the ingress LSR to the last LSR at the point of failure Fig. 1.

An important drawback in this scheme is the delay involved in detecting the first packet that is sent back from a point of failure to ingress LSR plus the delay for the subsequent packets sent along the broken LSP to return to the ingress LSR. Further, this approach also introduces data packet disordering during the LSP rerouting process. This is because once a fault is detected, the ingress node merges the newly incoming traffic and the packets coming back from the point of failure when sending them along the alternate path.

IV. Proposed mechanism

Our proposal follows the principle described in [5] for setting both an alternative LSP and a backward LSP. Changing to an alternative path is the responsibility of the routing control

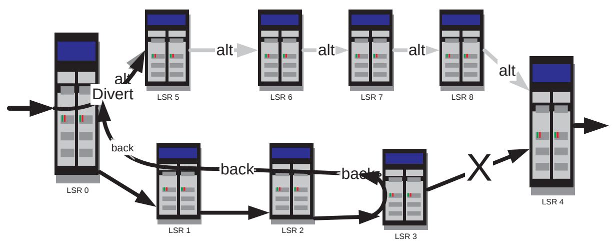

Fig. 1. Scheme for alternative LSP to handle fast rerouting during the restoration period. (back: backwards LSP; alt: alternative LSP)

process (e.g Label Distribution Protocol (LDP)[8] , extension of RSVP protocol [9]) and hence, is not within the scope of our study. The motivation of our study is to overcome the drawbacks of Haskin's [5] proposal with respect to round-trip delay and packet disorder.

In our proposal, when a fault is detected by a LSR, a switchover procedure is initiated and the packets are sent back via the backward LSP. As soon as each upstream node on the backward LSP detects these packets, they start storing the incoming packets (on the primary or protected path), in a local buffer. This avoids the unnecesarry forwarding of packets along the broken LSP. Furthermore, the last packet forwarded before initiating storing is tagged in order to be identified on its way back. By doing this, we are able to preserve the ordering of packets when it is time for each intermediate node to send back its stored packets. We use one of the Exp field bits of the MPLS label stack [10] for the purpose of tagging and thereby avoid any overheads.

Each LSR - on the backward LSP - successively sends back its stored packets when it receives its tagged packet. When all packets return to the ingress LSR (i.e. the ingress LSR receives its tagged packet) and have been rerouted to the alternative LSP, the restoration period terminates. The packets stored during this time in the ingress LSR, along with all new incoming packets are now sent via the alternative LSP. Note that global ordering of packets is preserved during the whole process.

The detailed algorithm along with the state machine diagram are presented in the next section.

V. Algorithm description

Fig. 2 presents the state machine diagram of proposed algorithm. The ingress LSR forwards packet to the alternative LSP while intermediate LSR forwards packets through the backwards LSP. Though the state machine diagram by itself is a

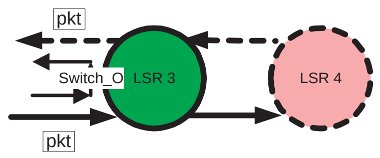

Fig. 3. FAULT_DETECT and Switchover

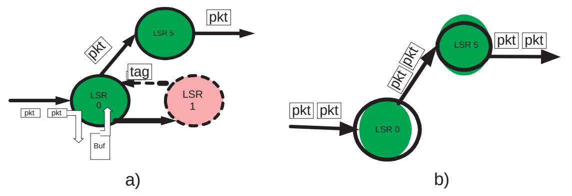

Fig. 5. a) SEND_BUFFER for ingress LSR. b)Restoration period terminates.

formal description, a detailed explanation of the process follows.

Once a failure along the protected LSP is detected, the protected LSR that detects the fault performs the switchover procedure (LSR3 in Fig. 3.) This procedure consists of a simple label swapping operation from protected LSP to backwards LSP for all packets with label corresponding to the protected LSP. As a result, the LSR that detected the link failure (LSR3) reroutes incoming traffic back by redirecting to the backwards LSP traversing the LSRs in the reverse direction of the protected LSP. The link status of the label information base forwarding table (LIB) of this LSR (<input_label, output_interface> corresponding to this forwarding entry table) is changed from NORMAL to FAULT_DETECT (Fig. 2).

As one can observe in Fig. 3, this switchover operation sends the packets to the upstream LSR. The intermediate upstream LSR, in this case LSR2 (Fig. 4a) receives these reversed packets from LSR3 through backwards LSP. At this moment, it changes the link status of the LIB entry of the protected LSP corresponding to this backward LSP to ALTERNATIVE_DETECT. Then, the first packet received from protected LSP sees this entry as ALTERNATIVE_DETECT. This indicates that there is a link problem somewhere in the protected LSP. So, this packet must be tagged as the last packet from this LSR (LSR2) and forwarded normally downstream and the LIB entry status must be changed from ALTERNATIVE_DETECT to STORE_BUFFER (Fig. 4a). The next packet in the protected LSP will be stored in the buffer because it will find the link status as STORE_BUFFER. This continues until the tagged packet is received through backwards LSP.

Packets from backwards LSP must be analyzed to look for the tagged packet. To know this, the LSR has to check if the tag bit of the received packet from backwards LSP is set (1) or not (0). If the comparison result is false the packet will be forwarded using normal swapping operation. Otherwise, it knows that no more packets are expected from backwards LSP. The tag bit in the label must be disabled (set to 0) and the packet is sent according the label swapping result as a normal packet. Moreover, it changes the status from STORE_BUFFER to SEND_BUFFER, and then when the buffer is empty, to FAULT_DETECT. Finally, the label associated with the protected LSP is removed. This process is repeated at every LSR until reaching the ingress LSR. Fig. 5 shows the opreration at the ingress LSR.

Our proposal avoids sending packets downstream once any intermediate LSR between the ingress LSR and faulty point detects packets on the backwards LSP. This reduces considerably the average delay time of packets travelling in the protected LSP during the detection of the fault in the distant LSR.

VI. Simulations and Results

The simulation tool used to evaluate the proposal is an extension of the network simulator (NS) for MPLS networks called MPLS network simulator (MNS) contributed by Gaeli Ahn and Woojik Chun [11][12]. The network simulation version used is NS-2.1b5. We use the CBR traffic flow and UDP agent with the following characteristics: packet size = 200 bytes, source rate= 400K, burst time=0 and idle time =0. The simulated scenario is the one shown in Fig. 1.

The simple network topology with a protected and alternative LSP is used. We extend the simple network topology for different number of intermediate LSR in the protected LSP giving place to different sizes of LSPs (i.e. LSP with 5, 10 and 15 LSRs) to analyze and compare the behaviour of both proposals.

We modified part of the MNS source code to satisfy our particular requirement for the simulation of both mechanisms (ours and Haskin's [5]). Note that the simulation platform for these proposals was the same in order to be able to compare the simulation results. We compared our results of the value of disordered and dropped packets during path restoration with the ones published in [13]. They are identical, thus validating our modified simulator.

Figure 6a shows the overall restoration period for both proposals for different number of LSR. Time is computed since

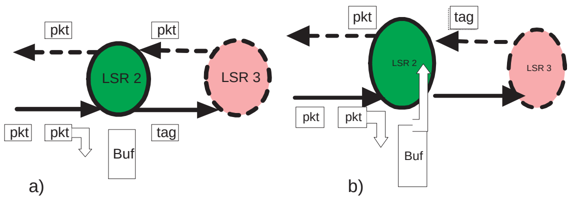

Fig. 4. In intermediate LSR a) ALTERNATIVE_DETECT and STORE_BUFFER b) Tagged packet received and SEND_BUFFR.

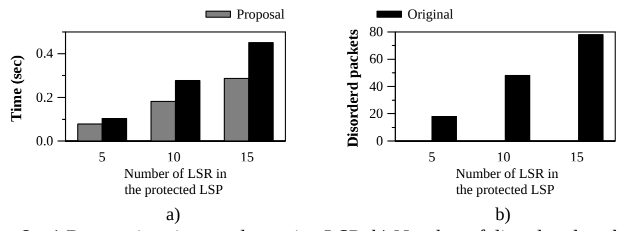

Fig. 6. a) Restoration time to alternative LSP. b) Number of disordered packets

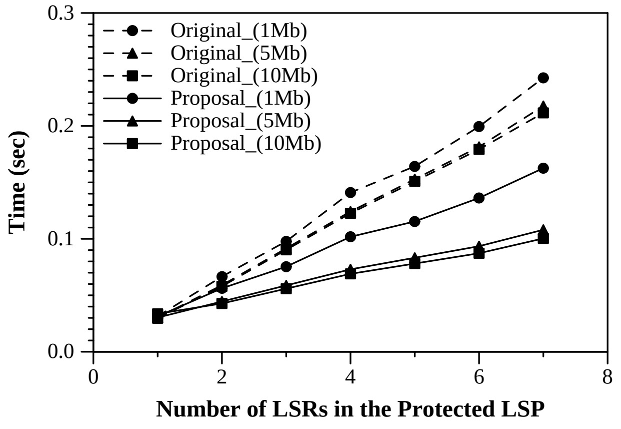

Fig. 7. Restotration delay for 200bytes packet size

the fault is detected until the protected LSP is completely eliminated. The proposed mechanism provides a significant improvement of the path restoration period. A reduction of about 24.12%, 34.05% and 36.37% for 5, 10, and 15 LSRs on the LSP respectively are achieved.

Figure 6b confirms that the proposed mechanism avoids packet disordering while the restoration is in process. In Fig. 7 we can observe the simulation results for the restoration time and different bandwidths given the same traffic. Fig. 8 varies the packet size for a given bandwidth (5Mb). In both cases, the number of intermediate LSRs were varied from 1 to 8. As can be seen from both figures, the reduction in restoration time is significantly better for the proposed mechanism for longer protected LSP.

Shortening the restoration period and the average packet delay during restoration, together with the fact of preserving packet sequence, minimizes the effect of a fault and leads to an improvement of the end-to-end performance.

VII. Conclusions and future work

This paper presents a mechanism to perform a fast rerouting of traffic in Multiprotocol Label Switching (MPLS) networks. We propose a method to avoid packet disorder and improve the average delay time during the restoration period. The proposed mechanism can be used for quality of service (QoS) provision. Once a given LSR detects congestion or a situation that leads to a Service Level Agreement (SLA) or QoS agreement being violated, it may start a fast reroute of a protected LSP that share the link.

An LSP rerouted due to congesion may experience a slight delay increase for a short period but no packets will be lost or disordered. Unlike the problem of failure in the link or node, the congestion problem gives us a time to manoeuvre the rerouting of packets towards the alternative path. To extend our mechanism to the congestion problem one only need to guarantee that the LSR be aware of it - just as in the case of link fault. If this condition is satisfied, we can divert the flow to the alternative path during congestion situation.

In summary our proposal has the following advantages:

1. Improves the average latency (average packet delay).
2. Avoids packet disorder.
3. Improves end-to-end performance (overall performance). and
4. Has a shorter restoration period than the original proposal (i.e. Fast network resources release).

Finally, the criteria for selecting alternative LSP for QoS provision and the extension of the proposed mechanism to avoid or reduce packet loss is left for further study.

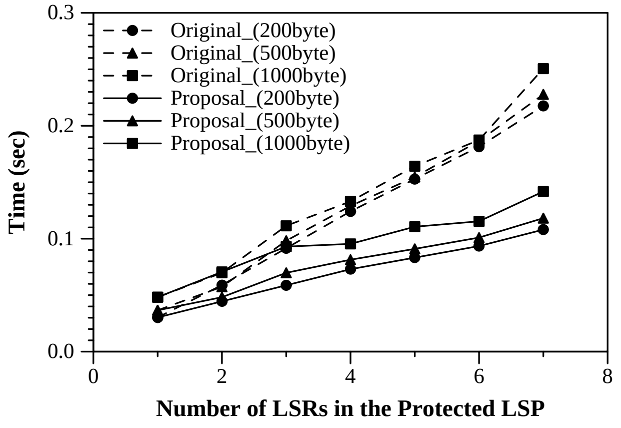

Fig. 8. Restotration delay for a 5Mbps LSP

References

[1] R. Callon, P. Doolan, N. Feldman, A. Fredette, G. Swallow, and A. Viswanathan, “A framework for multiprotocol label switching,” *Internet draft<draft-ietf-mpls-framework-05.txt>,*, September 1999.

[2] E. Rosen, A. Viswanathan, and R. Callon, “Multiprotocol label switching architecture,” *RFC 3031,*, January 2001.

[3] D. Awduche, J. Malcolm, J. Agogbua, M. O’Dell, and J. McManus, “Requirements for traffic engineering over mpls,” *RFC 2702,*, September 1999.

[4] V. Sharma, Ben-Mack Crane, S. Makam, K. Owens, C. Huang, F. Hellstrand, J. Weil, L. Andersson, B. Jamoussi, B. Cain, S. Civanlar, and A. Chiu, “Framework for mpls-based recovery,” *Internet draft<draft-ietf-mpls-recovery-frmwrk-01.txt>,*, November 2000.

[5] D. Haskin and R. Krishnan, “A method for setting an alternative label switched paths to handle fast reroute,” *Internet draft<draft-haskin-mpls-fast-reroute-05.txt>*, November 2000.

[6] S.Makam, V.Sharma, K.Owens, and C.Huang, “Protection/restoration of mpls networks,” *Internet draft<draft-makam-mpls-protection-00.txt>,*, October 1999.

[7] G. Swallow, “Mpls advantages for traffic engineering,” *IEEE Comunication Magazine,*, pp. 54–57, December 1999.

[8] L. Andersson, P. Doolan, N. Feldman, A. Fredette, and B. Thoma, “Ldp specification,” *RFC 3036,*, January 2001.

[9] Daniel O. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, and G. Swallow, “Rsvp-te: Extensions to rsvp for lsp tunnels,” *Internet draft<draft-ietf-mpls-rsvp-lsp-tunnel-07.txt>,*, August 2000.

[10] E. Rosen, D. Tappan, G. Fedorkow, Y. Rekhter, D. Farinacci, T. Li, and A. Conta, “Mpls label stack encoding,” *RFC 3032,*, January 2001.

[11] A. Gaeil and C. Woojik, “Design and implementation of mpls network simulator (mns) supporting qos,” *15th International Conference on Information Networking,*, January 2001.

[12] A. Gaeil and C. Woojik, “Design and implementation of mpls network simulator (mns) supporting ldp and cr-ldp,” *proceedings of the IEEE International Conference on Networks (ICON’00),*, September 2000.

[13] A. Gaeil and C. Woojik, “Simulator for mpls path restoration and performance evaluation,” *http://flower.ce.cnu.ac.kr/˜ fog1/mns/index.htm see path protection/restoration*, April 2001.