Efficient Distributed Solution for MPLS Fast Reroute

Dongmei Wang and Guangzhi Li

AT&T Labs- Research,
180 Park Avenue, Florham Park, NJ 07932
{mei,gli}@research.att.com

As service providers move more applications to their IP/MPLS (Multiple Protocol Label Switching) networks, rapid restoration upon failure becomes more and more crucial. Recently MPLS fast reroute has attracted lots of attention as it was designed to meet the needs of real-time applications, such as voice over IP. MPLS fast reroute achieves rapid restoration by computing and signaling backup label switched paths (LSP) in advance and re-directing traffic as close to failure point as possible. To provide a guarantee of failure restoration, extra bandwidth has to be reserved on backup LSPs. To improve the bandwidth utilization, path-merging technique was proposed to allow bandwidth sharing on common links among a service LSP and its backup LSPs. However, the sharing is very limited. In this paper, we provide efficient distributed solution, which would allow much broader bandwidth sharing among any backup LSPs from different service LSPs. We also propose an efficient algorithm for backup path selection to further increase the bandwidth sharing. The associated signaling extension for additional information distribution and collection is provided. To evaluate our solution, we compare its performance with the MPLS fast reroute proposal in IETF via simulation. The key figure-of-merit for restoration capacity efficiency is restoration overbuild, i.e., the ratio of restoration capacity to service capacity. Our simulation results show that our distributed solution reduces restoration overbuild from 2.5 to 1, and our optimized backup path selection further reduces restoration overbuild to about 0.5.

1 INTRODUCTION

As service providers move more applications to their IP/MPLS (Multiple Protocol Label Switching [1]) backbone networks, such as voice over IP and online games, et. al., rapid restoration upon failure becomes more and more crucial. Recently MPLS fast reroute has attracted lots of attention as it enables the re-direction of traffic onto backup LSPs (label switched path) within 50 milliseconds in the event of a failure [2]. Today, Internet Engineering Task Force (IETF) is in the process of standardizing Internet draft [2]. The Internet draft [2] addresses the necessary signaling extensions for supporting MPLS fast reroute, and proposes using path merging technique to share the bandwidth on common links among a service LSP and its backup LSPs. However, no solution is provided to solve the issues: how to share the bandwidth among any backup LSPs from different service LSPs, and how to select the backup LSPs to maximize the backup capacity sharing.

There has been a great deal of work addressing restoration functionality and schemes in IP/MPLS networks [3,4,5,6,11,12,13,14,20,22,24] as well as how to manage restoration capacity and select optimized restoration paths without centralized server [7,8,9,10,23]. However, all of them deal with path-based end-to-end or segment-based restoration [21,22,24] except [13,14] addressing routing protocol fast re-convergence. To the best of our knowledge, we have not seen any published paper to address bandwidth sharing among any backup LSPs from different service LSPs, and how to select the backup LSPs to maximize the backup sharing for MPLS fast reroute. Since the backup LSPs are pre-established, even though they do not consume bandwidth until failure happens, the network has to reserve enough restoration bandwidth to guarantee the LSP restoration upon failure. Without bandwidth sharing among backup LSPs for different service LSPs, and without an efficient backup LSP selection algorithm, the network would reserve much more bandwidth on its links than necessary.

In this paper, we propose two schemes to achieve and maximize the bandwidth sharing among any backup LSPs from different service LSPs for MPLS fast reroute. The first scheme--an efficient distributed bandwidth management can be implemented with nearly zero effort since it does not require any routing/signaling extensions. With our proposed array operation at each node, the backup capacity sharing at each link is independent of the backup LSP selection algorithm. The second scheme--an efficient backup LSP selection algorithm requires extending signaling protocol to distribute and collect some additional routing information. But the signaling overhead is very small. As many other bandwidth sharing schemes [8,10,20,22,23], our solution is based on the observation that the restoration bandwidth can be shared by multiple backup LSPs so long as their protected service LSP path segments do not fail at the same time. Our proposals have no confliction with the IETF MPLS fast reroute draft [2], but rather enhancements to it.

To demonstrate the efficiency of our proposals, we evaluate the bandwidth efficiencies of our approaches and the IETF MPLS fast reroute Internet draft [2] via simulation on two networks: a US intercity backbone network and a Toronto metropolitan network from [12], here we refer the Internet draft [2] as the baseline. The results show that our two enhancements reduce the amount of reserved restoration bandwidth or overbuild dramatically.

The paper is organized as follows. Section 2 summarizes MPLS fast reroute scheme and discusses its bandwidth management issues. Section 3 describes our first enhancement and gives an operational overview of our proposed bandwidth management method. Section 4 presents our second enhancement of efficient backup path selection algorithm and introduces new signaling messages between neighboring LSRs. Section 5 and section 6 present the simulation scenarios and performance results.

2 MPLS FAST REROUTE

2.1 Baseline

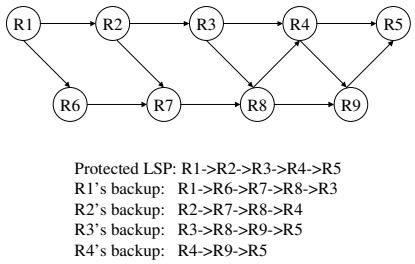

Figure 1: MPLS fast reroute example

IETF Internet draft [2] defines two MPLS fast reroute methods: one-to-one backup method and facility backup method. The one-to-one backup method creates detour LSPs for each protected service LSP at each potential point of failure, such as link failure or LSR failure. The facility backup method creates a bypass tunnel to protect a potential failure point. LSPs using the same protected facility will be protected via the same bypass tunnel. Thus, facility backup method can be pre-setup around potential failure point in the MPLS network and the bypass tunnels can be pre-designed in a centralized system to reduce network restoration capacity. However, the one-to-one backup method has to establish/delete backup LSPs on time as protected LSP comes and goes in a distributed MPLS network. In this paper, we only consider one-to-one backup method and propose mechanisms to maximize backup LSP capacity sharing on common links.

Figure 1 is an example from IETF Internet draft [2] illustrating one-to-one backup method. As [2] pointed out, to fully protect an LSP that traverses N nodes, there could be as many as N-1 backup paths. To minimize the number of LSPs in the network, it is desirable to merge a backup LSP to its service LSP when feasible. When a backup LSP intersects its service LSP at a LSR with the same outgoing interface, it will be merged. When two backup paths for the same protected service LSP travel a common link in the same direction, the required bandwidth can be shared since they are protecting different failure scenarios. When a failure occurs along the service LSP, the upstream LSR close to the failure point detects the failure and re-directs the traffic onto the local backup LSP. For example, if LSR R3 fails, R2 will detect the failure and redirect traffic along R2's backup path. Now the traffic will travel along R1-> R2->R7->R8->R4->R5. Since R2 cannot distinguish the failure between link R2->R3 and LSR R3, R2's backup usually excludes R3. Although backup LSPs do not consume bandwidth before failure, the network needs to reserve enough bandwidth along backup LSP links to guarantee 100% failure restoration. Using LSP merging technique, the bandwidth for service LSP and its backup LSPs can be shared along common links. The bandwidth for different backup LSPs of the same service LSP can also be shared along common links. But no solution is provided for backup LSP selection and bandwidth sharing among any backup LSPs from different service LSPs in [2] as we pointed out before. There is no surprising since Internet draft [2] is mainly focus-

ing on MPLS fast reroute definition and the signaling procedure for backup LSP creation/deletion. It leaves the algorithm of selecting the backup LSPs to maximize backup sharing to carriers/vendors for innovation and value-added enhancements.

2.2 Design Overview

Assume an MPLS network represented by a graph G=(V,E), where V is the set of network nodes[1] (e.g., label switch routers) and E is the set of network links. Protected LSP requests originating at clients arrive at the source nodes and they are established or deleted in a distributed manner, via signaling among the nodes. For each protected LSP request, the source node needs to compute a service path *Ps* and each node (say node *k*) except the destination node along the service path needs to compute a backup path *Pr(k)* in the network. The backup path *Pr(k)* protects failures of *k*'s immediate downstream node and immediate downstream link. Here *k*'s immediate downstream node is the next node of *k* along *Ps* and *k*'s immediate downstream link is the link between *k* and its immediate downstream node. In general, multiple service LSPs may share a common network resource, hence a single network resource failure, such as link failure or node failure, can cause multiple service LSPs to fail simultaneously. The design objectives for *Ps* and *Pr(k)* are as follows: (1) *Ps* and *Pr(k)* should be *k*'s immediate downstream node disjoint. (2) The required bandwidth associated with different *Pr(k)* of *Ps* should be shared if they share a common link. (3) Backup LSPs from different service LSPs should share bandwidth on common links if their protected service path failure points are not subject to simultaneous failure. (4) Enough bandwidth must be reserved on all links in the network such that for any link/node failure, there is enough bandwidth to restore all affected service LSPs. (5) Total bandwidth reserved for restoration over all links should be minimized. In MPLS networks, a link state routing protocol, such as Open Shortest Path First protocol (OSPF) or Intermediate System-Intermediate System protocol (IS-IS), is used to distribute network topology and resource information. The network resource information includes reserved bandwidth, available bandwidth, and administrative weights on each unidirectional link.

3 ENHANCEMENT I

3.1 Bandwidth Management Mechanism

Our bandwidth management mechanism provides a solution to book-keep the bandwidth reserved for restoration on each unidirectional link in a distributed way such that there is only necessary but sufficient bandwidth reserved on each link. For each link, there is bandwidth on both directions. Since each link connects two nodes, we

[1] In this paper, we use node, router, LSR interchangeably.

call the direction from smaller node id to larger node id as "up" direction and the reverse direction as "down" direction. Then the node with smaller/larger id is responsible for the bandwidth information of "up"/"down" direction. For each link direction, the responsible node maintains a local array *failother[f]*, where *f* is any possible failure points, denoted as set *F*. In this paper, *f* ranges over any single link and single node, which means $F=\{E\}\cup\{V\}$. *failother[f]* is the amount of bandwidth required on this link direction to restore all failed LSPs if failure *f* occurs. Then, reserving a bandwidth equal to $R = max\ \{Failother[f]: f\in F\}$ ensures that there are sufficient and necessary bandwidth reserved to protect against any single failure. *R* is distributed to all other network nodes using the routing protocol. During the lifecycle of an LSP, the operations include creation and deletion. We describe proposed signaling procedures for updating the local data structures during LSP creation and deletion below.

3.2 LSP Creation

When one source node receives a protected LSP request, it computes a service path *Ps* using constrained shortest path first algorithm (CSPF) with administrative weights and available bandwidth of each unidirectional link. In this paper, we assume the use of the RSVP Traffic Engineering (RSVP-TE) extensions [17] which allow signaling messages to be explicitly routed from the source to the destination. Similar extensions are required for constraint-based routing label distribution protocol (CR-LDP) [18].

LSP creation involves an initial signaling message PATH from the source node to the destination node with admission control along the computed service path, then a capacity reservation message RESV is returned from the destination node to the source node to finish the service path establishment. Upon receiving the RESV message, each node, say *k*, along the service path selects a local repair backup path to the destination node by excluding *k*'s immediate downstream node [2]. Then the node sends a signaling message PATH along the selected backup path to establish the backup LSP and reserve the shared restoration bandwidth. This PATH message will include *k*'s immediate downstream link and k's immediate downstream node information, which is used to maintain shared reservations at each unidirectional link along the backup path. The responsible node of unidirectional link *e* along the backup path updates the array *failother(e)* as follows: $failother(e)[f] \leftarrow failother(e)[f] + b$ where *f* are the immediate downstream link/node along the service path and *b* is the requested bandwidth of the protected LSP.

The implementation of this approach requires a consistent assignment of network link ids and node ids by all network nodes. Since link state routing gives each node a consistent view of the network topology, each link can be indexed by hashing the node ids of the two end points of the link in a standardized way. Another possible solution is for the network provider to provision a globally unique link id for each link.

3.3 LSP Deletion

When a source node receives a LSP deletion request, the network must delete the service LSP and its backup LSPs, and release the reserved bandwidth for shared backup capacity. Using RSVP-TE, the source node sends one PATHTEAR message to the destination node along service path. Each node, say k, along the service path also needs to send one PATHTEAR message along each backup path. Those PATHTEAR messages along the backup paths should include their immediate downstream link/node information such that the responsible node of each unidirectional link e along the backup path update the array *failother(e)* as follows: $failother(e)[f] \leftarrow failother(e)[f] - b$, where f is the immediate downstream link/node along the service path and b is the required LSP bandwidth. Then the reserved bandwidth $R = max\{failother(e)[f]: f \in F\}$ will be updated.

In summary, the updating of *failother* arrays does not require any additional routing and signaling messages. All it needs is to include the immediate downstream node/link information in the PATH and PATHTEAR messages for backup LSPs. Therefore, the proposed mechanism is very easy to implement in real MPLS networks with fast re-route scheme.

4 ENHANCEMENT II

4.1 A Centralized Algorithm with Complete Information

Paper [8] classified the routing information model as minimal information, partial information and complete information in end-to-end restoration. It found that complete information model is able to achieve optimized backup path selection but the complete information dissemination may not be scalable. Thus the complete information is only available for centralized server. In this enhancement, we first describe a centralized algorithm applying complete information to MPLS fast reroute, then we develop a method to realize this algorithm in distributed fashion without complete information flooding in the entire network.

The basic idea of our centralized algorithm is to select the backup LSPs over the links with more sharable bandwidths to reduce the additional required restoration bandwidth. To obtain the sharable bandwidth information, we define a matrix *failneed* from complete information in [8], where *failneed[f,e]* represents the amount of traffic that will be rerouted on unidirectional link e when failure f occurs. Then the sharable restoration bandwidth for a failure group Fg on unidirectional link e would be: $r(e) = R[e] - max\{failneed[f,e], f \in Fg\}$, where $R[e]$ is the reserved bandwidth on link e and provided via routing protocol. In MPLS fast reroute, after the service path Ps is selected, we need to select N-1 backup paths starting from each node along the service path except the destination node, where N is the number of nodes along Ps. Let $Pr(k)$ be the backup path starting from node k. Then $Pr(k)$ only requires disjoint from k's immediate downstream link/node along the service path Ps, which are denoted as

DL(k) and *DR(k)* respectively. Then the sharable restoration bandwidth on each unidirectional link *e* for failure group $Fg = \{DR(k),DL(k)\}$ would be: $r(e) = R[e] - \max\{failneed[f,e], f \in \{DR(k),DL(k)\}\}$. Next, we reset the link weights according to a function of *r(e)* (see details in section 4.3) and select *Pr(k)* as the shortest path with new link weights. To implement this algorithm on each node, the real challenge is how to collect the information of $\max\{failneed[f,e], f \in \{DR(k),DL(k)\}\}$. Note that each node does not need the complete matrix *failneed*, but the rows corresponding to its immediate downstream link/node failures. Here we propose using signaling protocol to collect the required information on each node. Our proposal keeps signaling overhead very small.

4.2 Signaling Extensions

As *failother* array in section 3.1, we propose that for each failure, a master node is responsible for maintaining the failure related information. For link failure, the master node could be the node terminating the link having the smaller node id. For node failure, the master node would be the node itself. Then the master node of each failure *f* along the service path maintains a *failself(f)* array. *failself(f)[e]* stores the bandwidth required on unidirectional link e to restore all affected LSPs if this failure f occurs. Note that *failself(f)* corresponds to the failure *f* row in the matrix *failneed* defined above.

We propose three new signaling messages between neighbor LSRs: TELL, ASK and REPLY for updating and obtaining the *failself* arrays.

- TELL: after a node selects or deletes its backup path, it will send a TELL message to its immediate downstream node along service path with backup path information.
- ASK: before a node selects its backup path, it sends a ASK message to its immediate downstream node for information.
- REPLY: when a node receives the ASK message, it replies the correct information back to the ask node.

The new message operates as follows: The ASK and REPLY messages are used for obtaining the *failself* array information. Before a node selects its backup path, it will send a ASK message to its immediate downstream node for *failself* array information for the immediate downstream node and/or the immediate downstream link. The REPLY message will reply the correct *failself* arrays of the immediate downstream node and the immediate downstream link information if it is the master node of the link.

The TELL message is used for updating *failself* arrays when a backup LSP is setup or teardown. After a node creates or deletes a backup LSP protecting the immediate downstream node/link failures, it sends a TELL message including backup LSP information to the immediate downstream node. Upon receiving the TELL message the immediate downstream node will update the *failselfs* array for itself and the immediate downstream link if it is the master node of the downstream link: $failself[e] \leftarrow failself[e] \pm b$, where *e* are the unidirectional links on the restoration LSP, *b* is the bandwidth of the LSP and "+" is for creation, "-" for deletion.

Since the *failself* arrays for the immediate downstream node/link failures are either installed in the immediate downstream node or itself, a node can easily obtain the *failself* array information needed to select the backup LSP path same as the centralized server. Therefore, we have successfully realized the centralized optimal backup path selection algorithm in a distributed fashion.

4.3 Algorithm in Detail

After a node k along the service path collects the *failself* array(s) from its immediate downstream node for immediate downstream link/node failures, it calculates a new array $T[e] = max(failself(DR(k))[e], failself(DL(k))[e])$, where e is for any unidirectional link. Then $T[e]$ is the maximum bandwidth needed on unidirectional link e if any one of the protected failures $DR(k)/DL(k)$ occurs, and the sharable backup capacity on e is $r[e] = R[e] - T[e]$, where R[e] is the reserved bandwidth on link e. After that, the node assigns a new weight to each unidirectional link e in the network:

$$w[e] = \begin{cases} (b - r[e]) \cdot W[e] & \text{if } b - r[e] > 0 \text{ and } e \notin \{DR\} \\ \varepsilon & \text{if } (b - r[e] \le 0 \text{ or } e \in \{DP\}) \text{ and } e \notin \{DR\} \\ \infty & \text{if } e \in \{DR\} \text{ or } A[e] < b - r[e] \end{cases}$$

where $\{DR\}$ means the set of links that adjacent to the immediate downstream node, which means the backup path should exclude this immediate downstream node, $\{DP\}$ means the set of links downstream from the immediate downstream node along the service path, $W[e]$ is the administrative weight on unidirectional link e, $A[e]$ is the available bandwidth on unidirectional link e, and ε is a very small positive number. Then shortest path algorithm is used to select the backup path using these weights. In this formula, the assigned new weight of ε is in favor of the selection of the links, which are the downstream path links for potential path merging and links with enough sharable restoration capacity. In addition, if a link does not have enough available capacity $A[e]$, it should be excluded from the backup path selection as well.

4.4 Scalability

The proposed scheme scales well due to two facts: (1) it does not require each LSR to maintain the global information of per LSP's service path and restoration paths. Only two arrays, i.e., *failself* and *failother*, are maintained at nodes independently. (2) it does not require each node to broadcast any additional messages in the entire network. The signaling overhead per backup LSP is confined between two adjacent nodes. Although we introduced three messages, in real implementation, only the TELL message needs to be added. The ASK and REPLY functionality can be embedded in the PATH and RESV messages as below. During the service LSP setup process, we use an optional field in PATH message to request the *failself* information. When RESV message comes back from destination to source, the *failself* array is carried in an optional field from downstream node to the immediate upstream node. Hence the only

signaling overhead in our proposal would be a single TELL message between two adjacent nodes per backup path.

5 STUDY METHODOLOGY

We evaluate the performance of our proposed approaches via simulation on two networks: the first is a US MPLS backbone network consisting of 18-PoPs (Points-of-Presence) and 30 links interconnected in a dual backbone router (BR) architecture[2] [19]; the second is the Toronto metropolitan network consisting 25 nodes and 45 links from paper [12]. In the first network, we aggregate all access routers (ARs) connecting to the same PoP into a virtual AR. Then each PoP consists of three nodes: two BRs and one AR. The AR-to-AR demands are generated using cumulative outgoing and incoming traffic from a demand forecast, i.e., all traffic starting and terminating at the AR respectively. Then we randomly select a demand according to source AR cumulative traffic outgoing probability and destination AR cumulative traffic incoming probability. In the second network, we do not have traffic forecast demands and thus assume uniform distribution traffic: randomly select two nodes for demand source and destination. The bandwidth of each demand is uniformly generated from an interval (10,100) Mbps for both networks. For simplicity, the MPLS link bandwidths are assumed to be deployed in units of OC48 channels (approx. X=2.5 Gbps). In our simulation, we assume traffic demands arrive at the network one by one, and never leave. In our simulation results, we generate 20 random runs and calculate the average value for each data point. We evaluate the performance of our proposals using restoration overbuild. We define $\alpha=\Sigma SC(i)$ and $\beta=\Sigma TC(i)$, where i varies over all links, SC(i) and TC(i) are the required number of OC48 channels for service only and for both service and restoration on link i respectively. Since service and restoration share the same bandwidth pool, it is possible SC(i)=TC(i) on some links. Then the restoration overbuild is calculated as $(\Sigma TC(i)- \Sigma SC(i))/ \Sigma SC(i)$.

6 SIMULATION RESULTS

We assume that the link capacities are set to infinity and LSP demands arrive one at a time. The service path is always routed along the shortest path. The backup paths will be selected using immediate downstream node disjoint shortest path to the destination node in baseline and our enhancement I while the backup paths in enhancement II are selected using our new link weight setting. Only merging technique is used in baseline scheme while distributed bandwidth management is used in both enhancements. After all LSPs are routed on the network, we calculate the restoration overbuild for each of the schemes.

---

[2] For router protection, IP/MPLS service providers may use two backbone routers at each PoP and each access router is dual-homed to the two backbone routers.

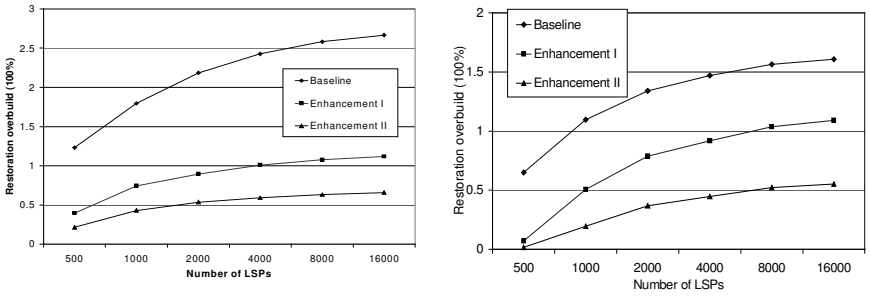

Figure 2: Network Restoration Overbuild

Figure 2 illustrates the average restoration overbuild for each of the three schemes. The left side sub-figure shows the restoration overbuilds for the first network and the right side sub-figure for the second network. Each algorithm is compared with the same traffic demands ranging from 500 to 16000 protected LSPs per network. Obviously, both enhancements require significantly fewer restoration overbuilds than the baseline scheme and our enhancement II requires the least for each demand set. Those conclusions are not surprising. Since baseline only uses merging technique and only shares bandwidth between the protected LSP service path and its backup paths, then each protected LSP reserves restoration bandwidth individually. Thus the baseline scheme reserves restoration bandwidth comparable to dedicated 1+1 restoration, which is very bandwidth consuming. Our enhancement I uses the default backup path selection but provides bandwidth sharing among any LSPs, it reserves the restoration bandwidth comparable to simple disjoint shortest path restoration with bandwidth sharing. Our enhancement II uses a little bit extra information to select the optimal backup path to maximize backup capacity sharing, it consumes the least restoration overbuild and it reserves the restoration bandwidth comparable to optimized shared mesh restoration with complete information [7,8,10].

We notice that the higher the traffic demands, the larger the restoration overbuilds in our simulation results. This is due to the restoration overbuild calculation formula. If the bandwidth in service channels is not fully consumed by service paths, it will be used for restoration paths. Fewer demands could leave more bandwidth on service channels for restoration paths. When the number of LSPs increases, the overbuild will stabilize. Note that enhancement I performs more than 100% and about 50% better than the baseline scheme in restoration overbuild for the first network and the second network respectively. Our enhancement II requires about close to half overbuild than enhancement I in both networks. Thus, there are strong advantages to use our proposed enhancements for MPLS fast reroute. All our enhancements use distributed operation and no centralized server is required. Also we noticed that the baseline scheme performs better in the second network than in the first one. One potential reason could be that the second network is denser than the first one. Using our enhancements, the restoration overbuilds are nearly the same for both networks. We also simulated blocking probability of the three fast reroute schemes on both networks and found out that both enhancements are able to reduce blocking probability significantly

from baseline and enhancements II always performs the best. Due to page limits, we left out the blocking probability simulation results.

7 CONCLUSION

We analyzed the problem of distributed bandwidth sharing and backup path selection for protected LSPs in MPLS fast reroute using one-to-one backup method. In particular, we proposed a distributed bandwidth sharing mechanism with implementation procedures. We also proposed a new distributed algorithm, which relies on limited routing information to achieve complete routing information results using simple signaling extensions between neighbor LSRs to distribute and collect extra information. We demonstrate that with small signaling overhead, we can use network capacity much more efficiently for MPLS fast reroute. We compared our new approaches against the IETF Internet draft [2] proposed solution. We conclude that the savings from our two enhancements are significant, while the implementation overhead is only marginal. Future work will focus on facility backup method of MPLS fast reroute.

References

[1]. E. Rosen et al. (ed.), "Multi-protocol Label Switching Architecture," IETF RFC 3031, 2001

[2] P. Pan, G. Swallow, and A. Atlas (Editors), "Fast Reroute Extensions to RSVP-TE for LSP Tunnels," IETF Internet Draft, draft-ietf-mpls-rsvp-lsp-fastreroute-05.txt, 2004, working in progress..

[3] S. Kini et al., "Shared Backup Label Switched Path Restoration", IETF Internet draft, work in progress, May 2001.

[4] V. Sharma and F. Hellstrand (ed.), "Framework for Multi-Protocol Label Switching MPLS-based Recovery," IETF RFC 3469, 2003.

[5] C. Gruber, "Bandwidth Requirements of MPLS Protection Mechanisms," 7th INFORMS TELECOM, Boca Raton, Florida, 2004.

[6] G. Li and D. Wang, "Efficient Restoration Capacity Design in MPLS networks," APCC/MDMC, Beijing, China, 2004.

[7] G. Li, D. Wang, C. Kalmanek and R. Doverpike, "Efficient Distributed Path Selection for Shared Restoration Connections", IEEE Infocom, New York, Vol. 1, pp.140-149, 2002.

[8] M. Kodaliam and T. Lakshman, "Dynamic Routing of Bandwidth Guaranteed Tunnels with Restoration", IEEE INFOCOM 2000.

[9] Y. Liu, D. Tipper, and P. Siripongwutikorn, "Approximating Optimal Space Capacity Allocation by Successive Survivable Routing," IEEE INFOCOM 2001.

[10] C. Qiao and D. Xum "Distributed Partial Information Management (DPIM) for Survivable Networks," INFOCOM 2002.

[11] R. Doverspike and J. Yates, "Challenges for MPLS in Optical Network Restoration," IEEE Communication Magazine, Feb. 2001.

[12] Y. Xiong and L. Mason, "Restoration Strategies and Spare Capacity Requirements in Self-Healing ATM Networks," IEEE/ACM Transactions on Networks, vol 7(1), February 1999.

[13] A. Basu and J. Riecke, "Stability Issues in OSPF Routing," SIGCOM 2001.

[14] C. Alaettinoglu, V. jacobson, and H. Yu, "Towards Millisecond IGP Convergence," Internet draft, draft-alaettinoglu-isis-convergence-00.txt, IETF, Nov. 2000.
[15] D. Katz et al., "Traffic Engineering (TE) Extensions to OSPF Version 2," RFC 3630, Sept. 2003.
[16] H. Smit et al., "Intermediate System to Intermediate System (IS-IS) Extensions for Traffic Engineering (TE)," RFC 3784, June 2004.
[17] D. Awduche et al, "RSVP-TE: Extensions to RSVP for LSP Tunnels," RFC3209, Dec. 2001.
[18] J. Ash et al, "LSP Modification Using CR-LDP," RFC 3214, Jan. 2002.
[19] G. Li, et al, "Detail Study of IP/ Re-configuable Optical Network Architectures," Broadnets 2004.
[20] P. Ho, J. Tapolcai, and H. Moufth, "On achieving optimal survivable routing for shared protection in survivable next-generation internet", IEEE Transaction on Reliability, Vol 53(2), June 2004, pp216-225.
[21] Pin-Han. Ho, "Segment Shared Protection in Mesh Communications Networks with Bandwidth Guaranteed Tunnels," to appear IEEE/ACM ToN.
[22] P. Ho and H. Mouftah, “Reconfiguration of spare capacity for MPLS-based recovery for the Internet Backbone networks,” IEEE/ACM ToN, Vol 12(1), Feb. 2004, pp 73-85.
[23] S. Han and G. Shin, "Fast restoration of real-time communication service from component failure in multi-hop networks," ACM SigComm 1997.
[24] D. Wang et al, “Efficient segment-by-segment restoration,” OFC 2004.