A Cost-Effective Protection and Restoration Mechanism for Ethernet-Based Networks: an Experiment Report

Yi Lei Chung-Horng Lung
Department of Systems and Computer Engineering
Carleton University, Ottawa, Canada

Anand Srinivasan
EION Inc.
Ottawa, Canada

Abstract

Protection and restoration at the physical layer is fast, but may require dedicated hardware. On the other hand, IP, although it does not rely on specific topology, is slow. Protection based on MPLS is effective for rerouting traffic to a pre-established backup LSP in the case of failures. Failure detection plays a dominant role in protection and restoration. This research presents an approach for addressing MPLS link and node protection by making use of auto-negotiation for Ethernet. The method is efficient to detect failures in an Ethernet-based network and can be easily adopted to support MPLS protection with low overhead. The experiment results show that traffic recovery can be achieved in sub-30ms.

Keywords -- MPLS, link/node protection, restoration, Ethernet, auto-negotiation

1. Introduction

Protection against and restoration after link and node failures can be performed at several layers in a network [5]. The mechanisms for protection and restoration can be placed at the network layer (IP layer), data link, or the physical layer. Protection/restoration at the network layer uses IGP protocols [1] such as OSPF, ISIS and RIP to maintain and update its routing table. In the case of a failure, the IGP protocol takes into account the topology change and re-computes the routing table by using the shortest path algorithm. When all the routing tables in the network are recomputed, the traffic traveling through the original route will be redirected through the new route. Even though this has been proven to be robust and survivable, the recovery coverage of this methodology is slow and can take a long time, from several seconds to minutes [6].

Another approach is to provide protection/restoration in the physical layer, where SONET is used for optical transmission. Automatic Protection Switching protocol (APS) [3] is used to switch over from a failed fiber to a protected fiber. APS involves a window of time of 10 ms for fault detection and can achieve a recovery in 50 ms [5]. However, this fast recovery is achieved at the expense of inefficient use of bandwidth and is typically limited to SONET/SDH ring-based systems [1].

Protection at the physical layer is fast but may require dedicated hardware, such as SONET protection ring. In addition, physical layer alarms are not always available [9]. Conversely, IP does not rely on specific topology, but it is slow. Detection with RSVP hello-based method usually is also slower than layer-2 alarm-based approach. MPLS, which is between the IP and link layers, supports recovery mechanisms that provide a trade-off between recovery speed and deployment cost [7]. MPLS, allows a Label Switch Path (LSP) to be set up before the traffic arrives to support fast recovery.

Ethernet is currently getting more popular due to its simplicity and lower cost. Ethernet is battling its way out of the enterprise networks and entering the metro areas or even the core of the network [8]. The *Metro Ethernet Forum* is supporting a technique that utilizes MPLS to enhance the resiliency of the LAN technology. One of the motions of the Metro Ethernet Forum is to develop the Ethernet-based metro Networks Protection for 50 ms restoration using MPLS [4].

This paper focuses on protection for Ethernet networks using MPLS. Not many actual reports on protection are available for Ethernet networks. The aim of the experimental research is to develop a lightweight, yet efficient method (<50ms) to effectively support MPLS protection and restoration. The method does not use signaling protocols, which introduce higher overhead and are more expensive. Rather, the approach takes advantage of a simple mechanism, auto-negotiation, designed for the Ethernet. This paper also presents concrete experiment results.

The paper is organized as follows. Section 2 discusses MPLS protection and restoration. Section 3 presents experiment results. Section 4 is the conclusions.

2. MPLS Protection and Restoration

In MPLS networks, several different methods have been proposed for detecting link failure. Basically, there are two categories for detecting failure: it can either be detected by the signaling protocols or by the physical layer. In the MPLS, RSVP-TE is the signaling protocol used to set up explicit LSP, and LDP is the protocol used to set up the hop-by-hop LSP dynamically. In both

protocols, a hello message is sent out periodically to indicate that the neighbor, LSR, is alive. The absence of a Hello message can be used as an indication of link failure.

The Hello message is an optional functionality in RSVP-TE. It is not necessary for every router to send out such messages. In LDP, it takes 15 seconds to detect the failure [10].

The protection strategy relies on pre-establishing a backup LSP for specific primary LSP(s). When a primary LSP fails, the PLR (point of local repair) detects the failure and notifies the forwarding engine correspondingly. The idea is not new. Link or node protection has been discussed in RFC2702 and the literature [9]. Osborne et al. [9] provided an excellent description of how link/node protection can be realized. Readers can refer to the book for detailed discussion.

The forwarding engine then switches the packets from the failed primary LSP over to the backup LSP. The experiments conducted in this research involve the development of a failure detection module, a failure notification module and a forwarding engine module. The detection module is responsible for failure detection. The failure notification module is responsible for generating a notification message and delegating the message to the forwarding engine. The forwarding engine module is responsible for switching the packets over and forwarding them along the appropriate LSPs.

2.1 Failure Detection

Different mechanisms can be used to detect a link failure. Hello messages, which are sent out periodically by the signal protocols such as LDP or RSVP-TE can be used as an indication of a link failure. SONET/SDH uses the loss of signal as an indication of link failure. Auto-Negotiation [2] uses Fast Link Pulse (FLP) signals to indicate a link failure. This research makes use of auto-negotiation and a polling mechanism to detect a link or node failure for Ethernet-based networks. The following sections introduce each mechanism of the link failure detection in detail.

Detection by Signal Protocol

RFC 3036 defines an LDP [10] hello message: "Discovery messages provide a mechanism whereby LSRs indicate their presence in a network by sending a Hello message periodically". "If the timer expires without the receipt of a matching Hello from the peer, LDP concludes that the peer no longer wishes to label switch using that label space for that link or that the peer has failed." The timeout is 15 seconds, meaning that it takes 15 seconds for the LDP to discover the link failure. The failure detection by the signal protocol is too long to be acceptable.

With regard to RSVP-TE, the hello interval is configurable and can be as small as 5ms. However, there are three main disadvantages with using RSVP-TE: 1) If the interval is small, e.g., 5ms, the Hello message will create unnecessary overhead. 2) RSVP-TE is running on the IP layer and uses the Ethernet driver to send out packets. The sending of the Hello message cannot be greater than the capacity of the driver. For detection, the lower the layer it is, the faster. 3) If there is heavy traffic (congestion) in the network, the hello message cannot be delivered in anything less than 17.5ms. The RSVP-TE will incorrectly suppose that the link is broken. 4) The Hello message is a completely optional parameter in RSVP-TE; not every node in the network supports a Hello message. If one of the nodes along the LSP does not support it, we cannot rely on the Hello message for failure detection.

Detection by the Physical Layer

In SONET/SDH [3], failure detection is triggered by the loss of the signal, which can be detected in 10ms. This fast failure detection means the SONET/SDH has the ability to perform the restoring connection in 50ms. However, this methodology can only be used for SONET/SDH.

Auto-Negotiation in Ethernet

Ethernet Auto-Negotiation [2] is a mechanism that monitors the interface when a connection is established to a network device. Auto-Negotiation provides connection interoperability between IEEE 802.3 LANs. It currently supports 10BASE-T, 10BASE-T Full Duplex, 100BASE-TX, 100BASE-TX Full Duplex, and 100BASE-T4. [2] Moreover, it detects the existing modes of the device at the other end of the wire. Auto-Negotiation uses Fast Link Pulse (FLP) signals, as shown in Figure 1. FLP signals are a modified version of the Normal Link Pulse (NLP) signals to verify link integrity.

FLP bursts occur at an interval of 16.8ms with a duration of 2ms. The FLP signal is encoded as a 16-bit word and is composed of 17 to 33 link pulses (identical to the link pulses used in 10BASE-T) to determine if a link has a valid connection.

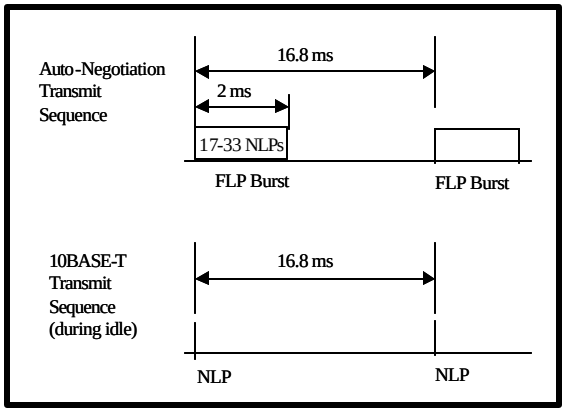

Figure 1. FLP Burst Timing

Failure Detection

In this research, failure detection is the responsibility of the PLR upstream of the protected link. Link and/or downstream node failures must be detected as early as possible in order to keep the total repair time low. To do so, the failure detection task keeps polling locally for any possible failure. Figure 2 illustrates the failure detection mechanism.

The Ethernet interface port, which is attached to the protected link, is periodically checked by this detection task. The periodic validation of the Ethernet interface status is expressed by a TRUE/FALSE Boolean value. A TRUE value implies that the link is still functioning, while a FALSE value implies that the link has failed. The FALSE validation result triggers the failure notification process.

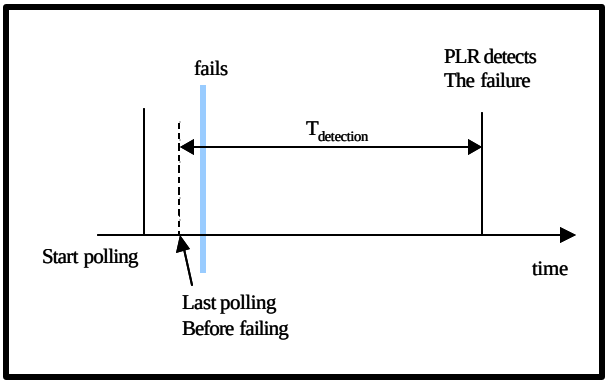

Figure 2. Link Failure Detection Timing

2.2 Failure Notification

Failure notification is accomplished by two consecutive steps: the construction of a failure notification message and the delegation of the notification message to the forwarding engine message queue.

The construction of the notification message requires that the message type and message contents be assigned. Apparently, the message type is set to the value FAILURE_NOTIFICATION. The message content is identified by the IP address of the Ethernet interface attached to the failed link. Once the notification message has been constructed, the notification module delivers it locally to the forwarding engine as a triggering step for the switchover process.

2.3 Forwarding Engine

Figure 3 shows the proposed MPLS forwarding Engine architecture. This architecture contains the IP module, DeviceDriver module, packetProcessor, MplsComLspManager mdule and the MplsComForwarder. In this architecture, the IP module and DeviceDriver are external modules to the forwarding engine. The IP module sends control messages that contain the labels' information to the PacketProcessor. The PacketProcessor performs differently for control packets or data packets. The PacketProcessor delegates the control messages to the MplsComLspManager module. The MplsComLspManager module decodes the received messages and updates the forwarding tables. The DeviceDriver sends the data packets to the PacketProcessor. The PacketProcessor delegates the data packets to the MplsComForwarder. The MplsComForwarder sends the packets out along the LSP.

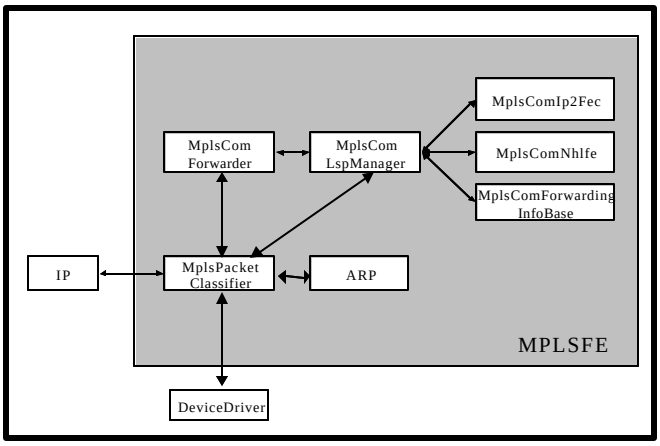

Figure 3. MPLS Forwarding Engine Architecture

3. Performance Evaluation

This chapter provides a performance analysis of the proposed MPLS protection design and implementation. The analysis evaluates the reliability, efficiency and credibility of the implementation. The evaluation includes link failure detection, failure notification and switchover to the backup LSP(s). The evaluation is based on successive experiments in a Linux environment providing fast Ethernet network adaptors. All experiments are processed using a recovery model proposed by IETF RFC 3469. The model considers timing criteria, which measures the elapsed time along the detection recovery process of a primary LSP failure.

Figure 4 describes the recovery model, which distinguishes three time zones. The first timing zone is failure detection time ($T_{detection}$). It ranges from the occurrence of a link failure to the time of the failure detected. The second timing zone is the notification time ($T_{notification}$). It ranges from the link failure notification to the beginning of the switchover (from the primary LSP to its backup LSP). The third timing zone is the switchover time ($T_{switchover}$). This ranges from the beginning of the switchover over to the end of the recovery process.

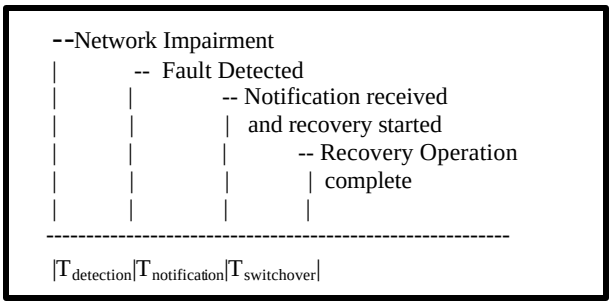

Figure 4. MPLS Recovery Cycle Model

3. Implementation and Experiments

The implemented MPLS protection is evaluated for a specific chosen link in a proposed environment. By assuming the Normal distribution scheme and choosing the level of significance as 0.95, the experiments achieve an average overall recovery-processing time of 29.542 ms. with a confidence interval of 28.94 ms ~30.10ms. The experiments are repeated for a large number of times. The resulting data is analyzed for the different sample sizes. The analysis shows that the confidence interval is very close for the different sample sizes. In this research, we report the collected result data for a given sample size of 15, which has half the deviation length of 0.6 with a precision of 2.2%. A precision of 2.2% is considered to be acceptable for this research.

3.1 Experiment Environment

The experiments were performed on a testbed environment of five routers running on Linux Operating systems. The Linux machine hosting the PLR node is an Intel (R) Celeron (R) CPU with a CPU speed 1715.189MHZ, while the vendor-id is Genuine Intel and the cache memory size is 8KB. The Fast Ethernet network interface card (NICs) installed is a RealTek RTL-8139. All links are full-duplex point-to-point 10/100BaseTX Ethernet cables.

3.2 Failure Detection Time

As per the recovery model definition, the link failure detection time is measured from the time the link is unplugged to the time when the PLR detects the link failure. In the link failure detection test, the failure is originated by manually unplugging the Ethernet cable connecting to a switch, which forms the assumed protected link. In the node failure experiment, however, the failure is originated by manually turning off the downstream router, which forms the assumed protected node. The implementation has a designated polling thread to detect the failure of the protected link. The time difference between the detection timestamp and the thread polling start time is the failure detection time $T_{detection}$. The first stage of the experiment ends by recording the failure detection time. The experiment is also repeated for the direct connection between two routers, the result is very close. This research assumes that the failure detection time for both cases is the same.

Since the testing result for the link failure detection time and node failure detection time is very close, we only present the case for link failure and protection. Figure 5 shows the distribution of the failure detection time recorded from the experimental results. Normal Distribution is assumed where the chosen level of significance is 0.95. The graph shows that the time varies between 25 to 30 ms. The average for the $T_{detection}$ is 27.02ms, with a confidence interval of 26.5ms~ 27.542ms.

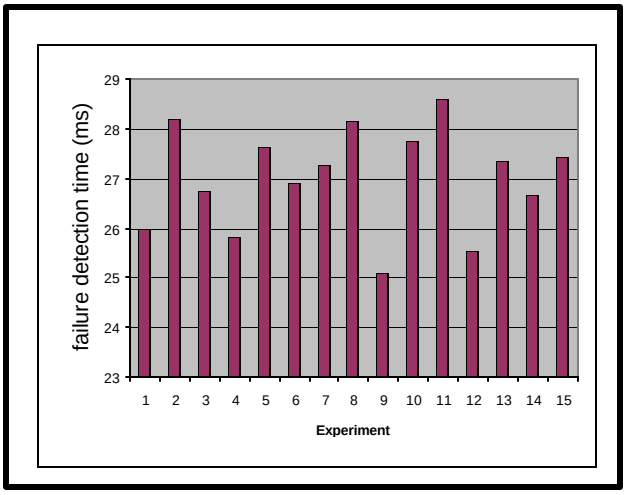

Figure 5. Failure Detection Time Distribution

3.3 Failure Notification Time

According to the recovery model definition, the failure notification time is measured from the time that the failure detection thread signals the notification, to the time when the forwarding engine receives the notification.

The failure detection thread generates a notification message and delegates it to the forwarding engine. The forwarding engine receives the notification message and issues a timestamp to record the arrival time of the message. The time difference between the failure detection timestamp and the notification arrival timestamp is the notification time $T_{notification}$.

The entire experiment (including detecting all timing zones throughout the recovery process) was repeated 15 times. Normal Distribution is assumed, where the chosen level of significance is 0.95. The time varies between 1.8 and 3.2 ms. The average for $T_{notification}$ is 2.5ms, where the resultant confidence interval is 2.376ms ~ 2.6244.

3.4 Switchover Time

According to the recovery model definition, the switchover time is the time taken to switch the primary

LSP to the backup one. The forwarding engine receives the notification message and begins updating the forwarding information table. At the end of the updating process, the time is recorded by generating an end-of-updating-process timestamp. The switchover time is measured from the arrival time of the notification message to the end-of-updating-process timestamp.

Again, Normal Distribution is assumed, where the chosen level of significance is 0.95. The switchover time varies from 19 to 26 us. The average for the $T_{switchover}$ is 21.467us, where the resulting confidence interval is 20.215us ~ 22.72us.

Table 1 illustrates the switchover time for the different numbers LSPs using simple linear search to exhaust the LSPs in case of a failure. As expected, the switchover time is slightly higher for a large number of LSPs. However, of the total recovery time, it is still the least time-consuming procedure (much less than 1ms).

Table 1 Switchover Time

Number of LSPs	Switchover Time (us)
1	21.47
10	27.25
100	112.71
200	180.42
300	211.13
400	243.17
500	282.71
600	347.29
700	411.26
800	458.63

3.5 Recovery Time

The total recovery time consists of three timing zones: failure detection time ($T_{detection}$), failure notification time ($T_{notification}$) and switchover time ($T_{switchover}$). The observations for each time experiment are recorded. The average of $T_{recovery}$ is 29.542. By assuming Normal Distribution and choosing the level of significance as 0.95, the Confidence Interval for the $T_{recovery}$ is 28.94ms ~ 30.14ms.

Apparently, the detection time is the bottleneck of the overall recovery time. This is concluded from Table 2, which shows the weight of the $T_{deteciton}$, $T_{notificaiton}$ and $T_{swithover}$ over total $T_{recovery}$. $T_{detection}$ represents 91.47% of the $T_{recovery}$. Any enhancement of the detection time will improve the performance of the recovery process.

Table 2 Recovery Time Weight Distribution

	Average time	Weight
Failure detection time	27.02 ms	91.47%
Notification time	2.5 ms	8.45%
Switchover time	0.0215 ms	0.08%
Total recovery time	29.54 ms	100%

4. Conclusions

This research presented experimental research to measure the time used for MPLS protection/restoration for Ethernet networks. An MPLS forwarding engine was implemented to support the experiment. Failure detection plays a crucial role in protection. We made use of an existing technique, auto-negotiation, which does not need any additional hardware or network layer protocols.

The experiment was evaluated on a simple Ethernet network environment for feasibility study. The experiment was repeated a number of times and it achieved an average of 30ms recovery time. The approach is simple enough to be deployed as an application program in the user space. The detection time can be reduced if the program is in the kernel space or is given higher priority. The switchover is performed using sequential search of the forwarding table for experiment purpose. Nevertheless, the total recovery time is still efficient, satisfactory, and reliable. Moreover, it can be used a reference point for Ethernet-based networks.

References

[1] Ayandeh, S., "Convergence of Protection and Restoration in Telecommunication Networks", *Photonic Network Communications*, 4:3/4, pp. 237-250, 2002.

[2] Becker, D., NWay Auto-Negotiation, 1995, Last visited Sept 2003, URL: http://www.scyld.com/expert/NWay.html#1.0

[3] Bellcore, GR-1230-Core, *SONET Bidirectional Line-Switched Ring Equipment Generic Criteria*, Issue 2, Nov 1995.

[4] Chen, N., "Defining and Delivering Protection for Metro Ethernet Networks", Last visited Sept. 2003, URL http: //www.supercommnews.com/wednesday/expers/metro.cfm

[5] Demmester, P. et al., "Resiliency in multiplayer networks", *IEEE Communications Mag.*, vol. 37, no.8, 1999, pp.70-76.

[6] Huang,C., Sharma, V., Owens, K., Makam, S, "Building reliable MPLS networks using a path protection mechanism", *IEEE Communications Magazine*, vol. 40, no. 3, March 2002, pp. 156 –162

[7] Ho, H., Mouftah, H., "SLSP: a new path protection scheme for the optical Internet", *Optical Fiber Communication Conf. and Exhibit*, March 2001.

[8] JUNOS Internet Software Configuration Guide, Last visited Sept. 2003, URL: https://www.juniper.net/techpubs/software/junos/junos57/swconfig57-mpls-apps/html/rsvp-config7.html

[9] Osborne, O., Simha, A., *Traffic Engineering with MPLS*, Cisco Press, July 2002.

[10] Wu, J., Montuno, D.Y., Mouftah, H., Wang, G., Dasylva, C., "Improving the reliability of the label distribution protocol", Local Computer Networks, 2001. *Proc. LCN 2001. 26th Annual IEEE Conf*, Nov. 2001, pp. 236 –242.

Acknowledgements:
This work was partly funded by the NCIT (National Capital Institute of Telecommunications), Ottawa, Canada. Thanks to M. Zaid, R. Crawhall of NCIT, and R. Munikoti, S. Kalaichelvan of EION Inc., for supporting this research.