A Comprehensive Fault-Tolerant Ethernet-Based IP Network Design

Jan Baranski
Electrical and Computer Engineering
Mississippi State University

Marshall Crocker
Electrical and Computer Engineering
Mississippi State University

Georgios Lazarou
Electrical and Computer Engineering
Mississippi State University

ABSTRACT

Using commonly available hardware and software, we present a proxy scheme for IP over Ethernet networks that provides a fault-tolerant solution without the need for modification of existing networking equipment. This type of fault-tolerant reconfigurable Ethernet-based proxy (FREP) is transparent to applications and provides full redundancy with minimal packet loss and fast reconfiguration times. A prototype implementation yielded a reconfiguration time of 1.55 sec and an almost instantaneous recovery time of < 0.02 sec. The proposed proxy scheme can be deployed in any 2-connected network and relies on a dynamic routing protocol to provide inter-network routing. Thus, we also present a comparison study of the RIP, OSPF, and EIGRP dynamic routing protocols while operating in conjunction with the FREP. The protocols were analyzed in their ability to provide fast convergence rates with a minimum trade-off in CPU utilization and network overhead. Our results indicate that all three protocols can be finely tuned to provide desired convergence rates and are well suited for operation in a simple ring-shaped Ethernet network. Convergence rates as low as 3.3 sec. were achieved with all three routing protocols.

Introduction

Inexpensive and widely available hardware has made Ethernet one of the most widely used physical layer technologies in today's IP networks [1]. Recently, the introduction of Ethernet over fiber (i.e. 100BASE-FX) has increased its appeal for use in CAN environments by providing additional cabling options. Fault tolerance, however, still remains a major issue. Networks built on Ethernet technology can typically be constructed using bus, star, or tree topologies, which inherently allow for uninterrupted network service during the addition, removal, or failure of network nodes within a local network. While dynamic routing protocols provide automatic reconfigurability at the inter-network level, the topologies of these networks do not, however, provide fail-safe connections to the backbone. This lack of fault tolerance prevents Ethernet-based IP networks from being used in situations that require a more reliable network infrastructure.

*This work was supported by the (US) Office of Naval Research (ONR) under Contract Number N00014-02-1-0623

Intra-Network Fault-Tolerance via a Proxy

A number of redundancy schemes for Ethernet networks have been designed and tested over the past few years [2, 3, 4]. Most of these designs, however, require modification of networking equipment (hubs, switches, NICs, etc.) or require software modification of the individual network nodes. We present an approach that focuses on a simple design mated with a specific, but flexible network topology. With our design, we have attempted to devise a "plug-in" solution to typical IP over Ethernet networks that provides transparent redundancy with a minimal traffic footprint. We have focused on a design that is capable of being applied to unmodified, off-the-shelf hardware and software.

In a star topology or tree topology-based network, a parent network is only accessible to its subnets via a single router. Some standards, such as the FDDI protocol, address this issue; these technologies, however, often require expensive hardware and complex configurations. An Ethernet-based IP network can also provide multiple connections to a parent network by mating the typical star topology to a central ring-shaped backbone as shown

in Fig. 1. Although the central ring is not a necessary component to the redundancy scheme, it is practical and one of the most widely used WAN backbone topologies. The central ring provides two routes to the parent network from each star-shaped subnet. This type of configuration is typically not useful to an unmodified IP over Ethernet network, however. In order to take advantage of this network architecture, our design employs a fault-tolerant reconfigurable Ethernet-based proxy (FREP), which allows traffic to be forwarded to a backup router in the event of a primary router failure. This solution provides a redundant connection to the backbone with only the addition of the FREP device and without any modification of individual network nodes. The need for one FREP device per network also makes this a scalable solution that can be deployed on an "as-needed" basis – FREPs can be added only to networks that require fault-tolerance, while other networks remain unchanged.

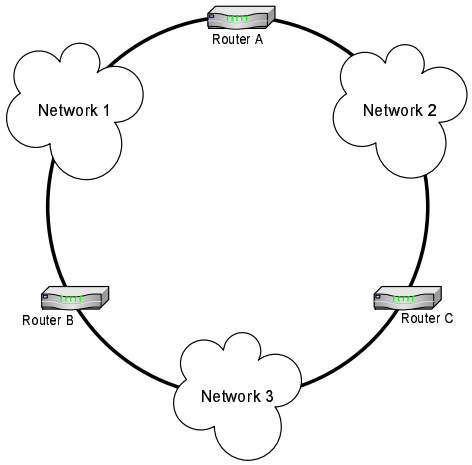

Figure 1. The analysis of the dynamic routing protocols was performed in a tested network that was constructed based on the star-ring topology.

Inter-Network Fault-tolerance via Dynamic Routing Protocols

In addition to providing fault-tolerance to individual subnets via the FREP, we also evaluated three dynamic routing protocols, Routing Information Protocol (RIP), Open Shortest Path First (OSPF), and Enhanced Interior Gateway Routing Protocol (EIGRP). Although previous work exists in which routing protocols have been compared [5, 6], no comparisons of the protocols' performances in response to a link failure in a small Ethernet-based network have been documented. Additionally, prior studies have generally employed simulation-based techniques and do not provide an experimental basis for the comparisons.

Each of the three above-mentioned dynamic routing protocols behaves differently upon detection of a failed link, and each has distinct advantages and disadvantages. Some comparisons of their performance and associated overhead traffic, however, can be made. Using test scripts and the FREP, we measured the duration of network outage after a link failure while running each of the routing protocols. These routing protocols, used in conjunction with the FREP, provide complete fault-tolerance to an Ethernet-based IP network.

FREP – Design Overview

The design of our transparent redundancy scheme is based on a normally configured IP network. In the typical IP over Ethernet network, nodes on a specific subnet are unable to communicate with any "outside" nodes when the router for that particular subnet becomes unreachable. Our solution uses the FREP to mimic the primary router for a particular subnet in case of a failure.

The FREP plays no role in packet routing during normal operation, although it repeatedly polls the primary router to ensure connectivity. This mode of operation is illustrated in Fig. 2. If the primary router is determined to be unreachable the FREP assumes the router's identity and transparently forwards all outbound packets to the backup router associated with that particular subnet. This "failover" mode of operation is shown in Fig. 3.

Inbound traffic must also be routed properly into the subnets during times of a primary router failure. A dynamic IP routing protocol ensures that the routers constantly possess an accurate view of the network and that inbound packets are properly routed around the failed link. This "transparent proxy" design scheme ensures proper traffic flow into and out of individual subnets during failover operation without any modification of existing network hardware or software.

While operating in failover mode, the FREP constantly "listens" for the availability of the primary router. Status of the connection to the primary router is monitored at the link layer to ensure minimal overhead. Once the primary router responds and is determined to be reachable again, the FREP relinquishes control back to the router and normal routing of traffic into and out of the subnet is resumed. By using a link layer protocol, we are also able to implement a "flap detection" mechanism, which allows the FREP to maintain control of the router's identity during times of state flapping[1].

[1] State flapping refers to the repeated changing of a state in a particular system. In this case, state flapping refers to the repeated changing state of the connection to a primary router. Thus, flap-detection allows us to maintain the primary router's identity and continue forwarding traffic

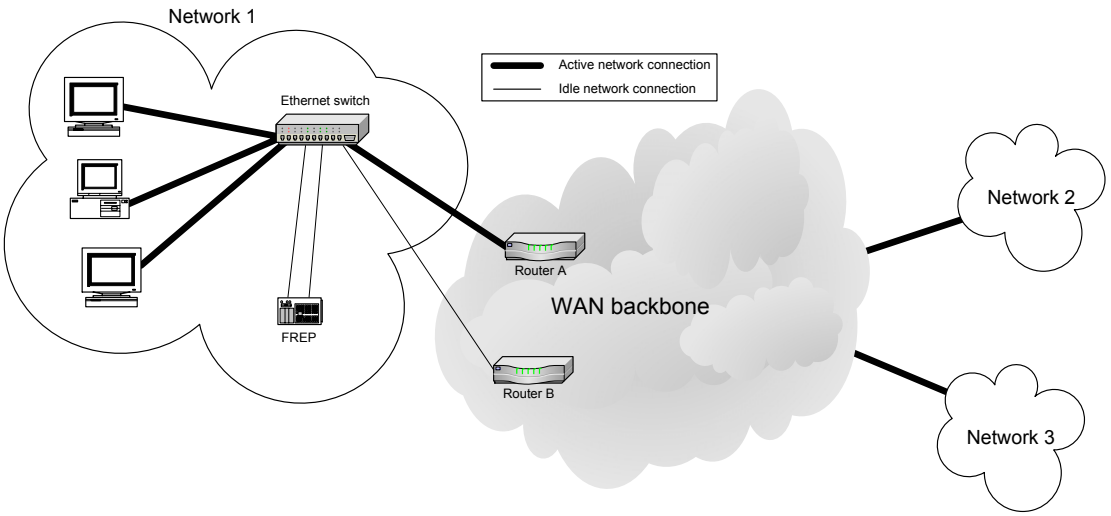

Figure 2. A subnet equipped with a FREP during typical operation. All traffic to/from the subnet is sent directly to the primary router, which is constantly polled by the FREP to determine connectivity status.

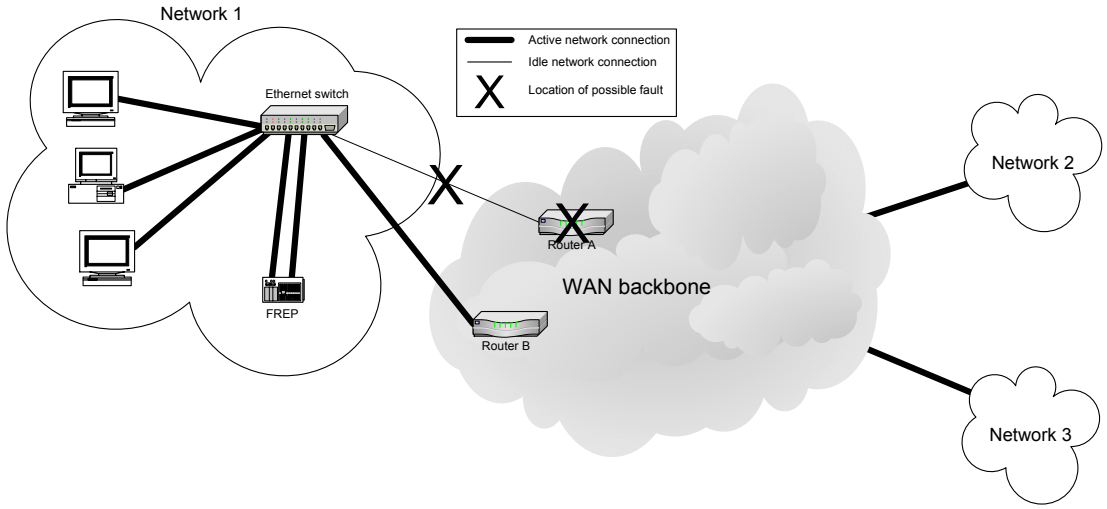

Figure 3. A subnet equipped with a FREP during failover operation. The primary router has become unreachable due to failure and the FREP has taken over its identity. All traffic sent from the nodes to the primary router is transparently forwarded to the backup router via the FREP. All inbound traffic enters the subnet through the backup router after reconfiguration in the backbone occurs via the dynamic routing protocol.

FREP – Implementation

The FREP was implemented using a Linux workstation with two network interface cards and custom software to perform the failover operations. The software was written in C and is responsible for the control of the FREP. The FREP begins by performing connectivity checks to the primary router at specified intervals. If the primary router is determined to be unreachable, the FREP enters a failover mode in which it forwards all traffic destined for the primary router to a backup router. While in failover mode, the FREP also monitors the status of the primary router and relinquishes control once it becomes reachable again.

Testing and implementation of the FREP took place in a testbed network consisting of three subnet/router pairs arranged in a hybrid star-ring topology. Netspec [7] software was used to place a load on the network during testing to more closely simulate a real-world network.

Connectivity Checks

The connectivity checks performed by the FREP consist of a `connect()` system call with a stream socket that

to a backup router until the primary router's connection has stabilized.

points to the echo port of the router. The `connect()` call is set to time out after a specified interval via synchronous I/O multiplexing by way of a `select()` system call. This method of attempting to establish a connection to the echo port ensures that the primary router is reachable via TCP traffic[2]. Most hardware and software routers available today are configured with the echo service disabled. This type of configuration yields a "Connection refused" error upon a connection attempt and can be used to determine whether the router is reachable. Thus, connectivity to the primary router is verified if the FREP is either allowed to establish a connection or if it receives the "Connection refused" error. If the connectivity check succeeds the software continues to perform checks at the user-specified interval. Otherwise, the FREP enters failover mode and its network interfaces are reconfigured to mimic the router.

"Failover" Mode

Once the switch to failover mode occurs, the *eth0* interface is assigned the IP address of the primary router. Since the primary router's IP address is now associated with a different hardware MAC address, this change must be propagated throughout the network. This operation is performed by broadcasting a gratuitous ARP packet[3]. This ensures that the ARP caches of all listening network nodes are updated with the new MAC address. At this point, all outbound packets sent to the primary router will be delivered directly to the FREP. Once the packets are received by the FREP, they are transparently forwarded to a backup router.

Recovery

While operating in failover mode, the FREP constantly monitors the status of the primary router. The monitoring operation is performed by two separate processes spawned via a `fork()` system call. The child process is responsible for sending out ARP queries for the MAC address of the router at specified intervals. The ARP queries are sent via the low level packet interface of the Linux kernel. The parent process listens for a response from the router using various methods from the pcap packet capture library [8]. Once a response is received, the child process is killed and control is returned to the caller. The FREP then relinquishes the router's IP address and resumes performing TCP connectivity checks.

[2]The FREP software can be modified to perform other types of checks such as ICMP or UDP pings. The TCP protocol was chosen to most closely match the traffic used in the connectivity check with the type of traffic that would typically be handled by a router.

[3]A gratuitous ARP packet is an ARP request or reply that forces all listening nodes to update their ARP cache. The packet sender and target addresses are both set to the IP address of the cache entry that is to be updated; the sender hardware address is set to the hardware address to which the entry should be updated. [9]

FREP – Performance

The performance of the FREP software and hardware is highly dependent upon the user defined parameters that are set in the program's configuration file. The following parameters can be changed by the user and have a direct effect on the operating characteristics of the FREP:

- Interval between TCP connectivity checks
- Timeout of TCP connectivity check
- Number of allowed failures for TCP connectivity check before a switch to failover mode occurs
- Interval between recovery checks (ARP requests for the primary router's MAC address)

By modifying these parameters the user has total control over the behavior of the FREP. If high availability, for example, is not an issue the interval between connectivity checks can be increased to a higher value. This setting prevents the FREP from frequently attempting to connect to the router, while still providing redundancy in the event of a failure. Thus, the trade-off that the user is faced with is response time versus network overhead. Although the overall network footprint of the FREP device is minimal, performing frequent connectivity checks on a busy network may not be desired.

Performance of the FREP was measured using the Tcpdump packet capture tool [10], which is available with most standard Linux distributions. The effects of a network failure and a recovery performed by the FREP device were observed using TCP, UDP, and ICMP traffic. All of these tests were performed using a 1 sec interval between TCP connectivity checks. It should be noted that increasing this parameter by a particular amount has the effect of increasing the minimum values reported here by that same amount.

The time required for the FREP to complete the reconfiguration operation was measured as the time between the initial loss of connectivity to the primary router and the time of arrival of the first packet of traffic at the FREP. This time was measured to be 1.55 sec and is representative of the time required by the FREP to forward traffic to the backup router from the initial time of failure. It was observed, however, that the apparent interruption of network service to the local subnet was significantly greater and a result of the dynamic routing protocol employed to perform inter-network routing. The tests described here were performed in a testbed network that consists of three subnet/router pairs as shown in Fig. 1. By scaling down

the network to a two router/subnet pair architecture, the 1.55 sec. reconfiguration time can be observed between any two nodes and is the "raw" reconfiguration rate of the FREP. When expanded to a larger network, however, the FREP becomes dependent upon the convergence rates for the dynamic routing protocol used in the backbone routers.

The maximum average delay during the recovery operation was measured to be 0.017 sec. When an ARP reply is received from the primary router during failover operation, the FREP software reconfigures a network interface to allow the router to regain control of the subnet. The initial destination of network packets, however, is ultimately controlled by the sending devices. Thus, after connectivity to the router has been re-established, the FREP continues to receive and forward packets until the sending devices update their ARP caches with the hardware MAC address of the router. As a result of this behavior, an almost instantaneous switchover occurs when a sending device updates its ARP cache and resumes sending data directly to the router. Recovery operation rate was determined to be independent of network topology or routing protocol. Thus, long convergence times of dynamic routing protocols will not affect the duration of unavailability of network connectivity during the FREP recovery operation.

A Comparison of Dynamic Routing Protocols – Methods and Results

The performance of the RIP, OSPF, and EIGRP protocols was analyzed in our testbed network with the aid of the FREP. By using the FREP to forward packets to the secondary router after a link failure, we were able to measure the convergence rate of the dynamic routing protocols. The measurements were made from an end-user point of view by calculating the duration of network outage using TCP traffic with the RIP protocol and ICMP traffic with OSPF and EIGRP. The link failures were simulated by disconnecting the primary router from the network and Cisco IOS C1700 Software version 12.2(11)T2 was used. Unless otherwise mentioned, the parameters of the routing protocols were left at their default values. The analyses and their corresponding results are outlined below.

RIP

The IOS software used in Cisco routers always performs packet switching when two equal paths to a particular destination exist. The `ip route-cache` command controls the use of the high-speed switching caches in the routers. By default, switching occurs on a per-destination basis with the router keeping a cache of the preferred route to a specific destination. When two equal paths exist to a particular destination and a non connection-oriented protocol such as ICMP or UDP is used, the IOS software alternates the path that the packets travel. When ICMP or UDP packets are forwarded to the backup router after a link failure, it instantly sends packets to their destination via an alternate route. Thus, it is not possible to accurately measure the convergence rate of RIP using one of these protocols in a ring-shaped topology. By establishing a TCP connection before the link failure occurs, however, we are able to prevent the router from using the alternate route until convergence of the RIP protocol takes place.

In order to test the performance of the RIP protocol, a Perl application that communicates with a server via the TCP protocol was used. The program consists of a client and server and is responsible for both the generation of traffic via a TCP stream and the measurement of network outages. Upon connection to the server, the client program performs a fork and spawns a child process. The child captures network traffic via the tcpdump utility and calculates inter-packet arrival times of traffic from the server. The parent process, meanwhile, maintains a TCP connection with the server by sending 2-byte packets to the server at a 0.01 sec. interval. The server also responds with an identical stream of traffic to produce a bi-directional stream. Thus, by establishing a TCP connection with a remote machine and measuring the inter-packet delay for incoming traffic, we are able to measure the duration of network outage after a link failure and prior to convergence of the RIP protocol.

The RIP protocol allows four timers to be controlled by the user: update, invalid, holddown, and flush. These timers control the behavior of the protocol and have a direct effect on performance. The update timer controls the rate at which routing updates are sent. The invalid timer specifies the interval of time from the last update after which a route is declared invalid. In default configurations, the invalid timer is typically set to six times the update interval. The holddown timer specifies how long the hold-down phase lasts after a route has been declared invalid. During the hold-down phase, the router will not process any additional updates it receives regarding the particular route. The flush timer controls how long a router waits after receiving the last update before removing a route from the routing table. The flush timer overrides the hold-down timer and can be set such that the hold-down phase is never entered. By adjusting these three parameters and using a TCP-based test program, we measured the performance of the RIP protocol. Twenty trials were performed for each test case and the results are shown in Table 1.

In all test cases, the hold-down and flush timers were

Table 1
AVERAGE DURATION OF NETWORK OUTAGE AFTER LINK FAILURE MEASURED VIA A TCP-BASED ANALYSIS OF THE RIP PROTOCOL

RIP Timers (sec.)				Outage Duration (sec.)
Update	Invalid	Holddown	Flush	
1	3	3	6	3.29
2	6	6	8	7.35
3	9	9	12	10.60
4	12	12	16	12.84
5	15	15	20	15.52

set to a value that would avoid the hold-down phase of the RIP protocol altogether. Additional data also showed that the update timer will affect the deviation of the network outage duration, and overhead associated with the TCP protocol will add additional delays. In all five cases, connectivity was re-established within 1.60 sec. of the invalid timer (before the expiration of the flush timer).

OSPF

A similar utility to the TCP-based version described above was written for ICMP-based testing of the OSPF and EIGRP protocols. Due to the nature of ICMP echoes, however, a separate client and server were not required. The program behaves identically to its TCP-based counterpart, with the exception of the type of traffic that is generated. The ICMP-based test program initiates a periodic ICMP ping at a rate of 1 packet/0.01 sec. It also captures the replies and calculates the duration of time between subsequent responses.

Attempts to use TCP-based tests with the OSPF and EIGRP protocols yielded results that matched the operation of the TCP retransmission timer (RTO). If a TCP connection was established prior to the link failure, the TCP protocol would make attempts to reestablish connectivity based on the RTO. Thus, regardless of when convergence of the routing protocol took place, the measured duration of network outage was always a result of the RTO. The RTO is calculated as described in [11]. This behavior forced us to use ICMP-based tests during analysis of the OSPF and EIGRP protocols.

Four timers exist that affect the performance of the OSPF protocol, the hello timer, dead timer, SPF delay timer, and SPF hold timer. The hello timer controls how often a router sends "hello" packets to any listening routers in the same routing area. The dead timer specifies the amount of time after receiving the last hello packet after which it declares its neighbor "dead." The SPF delay timer specifies the amount of time that the router waits before performing the SPF calculation after receiving a routing update. Finally, the SPF hold timer defines the amount of time a router waits between performing subsequent SPF calculations. The SPF delay and SPF hold timers were both set to 0 during the analysis. In a small network, these two timers can typically be set to a low value because large SPF calculations will not be performed often. Thus, we adjusted the hello timer and the dead timer to measure the convergence rates of the OSPF protocol. Twenty trials were performed for each test case, and results of the analysis are shown in Table 2.

Table 2
AVERAGE DURATION OF NETWORK OUTAGE AFTER LINK FAILURE MEASURED VIA AN ICMP-BASED ANALYSIS OF THE OSPF PROTOCOL

OSPF Timers (sec.)[1]		Outage Duration (sec.)
Hello	Dead	
1	3	2.94
1	5	5.01
1	10	10.04
1	15	14.86
5	20	18.46

[1] The SPF delay and SPF hold timers were set to 0 sec. for all test cases.

In all test cases, connectivity was reestablished within 1.54 sec. of the dead timer. OSPF, however, does not cache multiple routes in the routing table. Thus, unlike RIP, the problematic routes were completely flushed from the routing table before packets were properly routed around the failed link. A higher hello interval also increased the deviation of the duration of network outage.

EIGRP

Our analysis of EIGRP was identical to the ICMP-based analysis performed on the OSPF protocol. Two parameters can be tuned by the user to control the performance of EIGRP, the hello interval and the hold time. The hello interval specifies the frequency at which a router sends "hello" packets. The hold time defines how long a router will wait before flushing a route from its table after receiving the last hello packet. By adjusting these two parameters, we were able to perform an analysis in which we measured the convergence rates of the protocol. Once again, twenty trials were performed, and the results are shown in Table 3.

Network connectivity was reestablished faster than the hold time in all test cases. Increasing the hello interval also increased the resolution of the duration of network outage as seen with the OSPF and RIP protocols.

Table 3
AVERAGE DURATION OF NETWORK OUTAGE AFTER LINK FAILURE MEASURED VIA AN ICMP-BASED ANALYSIS OF THE EIGRP PROTOCOL

EIGRP Parameters		Outage Duration (sec.)
Hello interval (sec.)	Hold time (sec.)	
1	3	2.73
1	5	4.58
1	10	9.51
5	15	12.65
5	20	17.54

A Comparison of Dynamic Routing Protocols – Analysis

RIP

Results of the RIP analysis show that the flush timer of the protocol does not need to expire in order for routing to resume after a link failure. Prior to a link failure, the router possesses two valid routes to a particular network; both routes are used in an alternating fashion when "fast switching" is enabled (default setting) in the IOS software. The router alternates use of the two paths based on a per-connection policy. Thus, if a particular TCP stream is using a route in which a link failure occurs, the router will not reroute traffic to the alternate route until the invalid timer for the failed route expires. The analysis results verify that only the invalid timer must expire, and the route does not need to be completely flushed before the alternate route will be effectively used for all traffic. Additionally, the update timer can be controlled by the user to provide a finer resolution of the convergence rate.

The routing information protocol (RIP) is one of the more widely used dynamic routing protocols today. It is a relatively simple Distance Vector protocol and is often used to provide inter-network routing for small WANs/LANs. A RIP version 2 has also been developed and adds advanced features to the protocol. RIP inherently supports an environment in which routing loops are present, which makes it well-suited for networks with a ring-shaped backbone topology. The additional overhead associated with an increased RIP update frequency must be considered carefully for low bandwidth networks, however. The maximum size of a RIP update is 1,080 bytes, and, given the worst case, will be broadcast every time the update timer expires. Although RIP is able to provide a fast convergence rate in ring-shaped networks, a significant amount of available bandwidth may be sacrificed to the routing protocol in such an environment. It should also be considered that in order to avoid the "count to infinity" problem, RIP distances are limited to 15 hops.

OSPF

Results from the ICMP-based analysis of the OSPF protocol indicate that convergence in a ring-shaped network takes place after the dead timer for a particular route expires. The duration of the outage can be further controlled via the hello timer. The effect of adjusting the OSPF hello timer is identical to that of adjusting the RIP update timer – the resolution of the duration of network outage is affected accordingly. Unlike RIP, OSPF routers do not broadcast their entire routing table each time the hello timer expires. OSPF sends only partial updates (Link State Advertisements – LSAs) when the effective topology of the network changes. Furthermore, the hello packets are minimal in size (68 bytes for a router with 20 neighbors) and do not contain any routing information. The OSPF protocol also supports multicasting, which allows it to keep from flooding the entire network with LSA traffic. A major trade-off to fault-tolerance, however, is the amount of processor time required by the protocol. Because OSPF is a Link State protocol that uses Djikstra's algorithm, shortest path first (SPF) calculations must be made each time an LSA is received. In a complex network, these calculations can become processor intensive and affect the ability of the router to properly route traffic. In a simple ring-shaped network, however, this issue does not pose a problem. Additionally, OSPF supports advanced features such as separate routing areas and variable length subnet masks, which allow it to scale well to large and complex networks.

EIGRP

As similarly observed with the OSPF and RIP protocols, results of our analysis show that the convergence rate of the EIGRP protocol can be tightly controlled by two parameters, the hold time and the hello interval. The hold time directly affects the duration of network outage, while the hello interval can be used to alter the possible range of convergence rates. EIGRP is a proprietary Cisco protocol that supports many advanced features, which allow it to scale well into complex modern networks. Additionally, EIGRP supports multicasting and uses small hello packets to maintain neighbor relationships. It is an advanced Distance Vector protocol and offers vast improvements over more traditional protocols such as RIP. EIGRP decreases convergence rates by employing a diffusing update algorithm (DUAL). Although the EIGRP protocol provided the fastest convergence rates and lower processor usage than OSPF, its main drawback during deployment is the requirement of Cisco routers.

Conclusion

By developing a passive proxy device (the FREP), we have shown that it is possible to inexpensively provide failover capabilities to Ethernet-based IP networks. Although this solution is not ideal for high-availability applications such as real-time networking, it does, however, offer a basic level of fault-tolerance for general purpose applications. We have shown that our FREP-based solution is able to reconfigure quicker than most standard deployments of the various dynamic routing protocols used today. With the inclusion of a comparison of the RIP, OSPF, and EIGRP protocols provides, we hope to provide a complete and inexpensive solution of providing fault-tolerance to existing IP over Ethernet networks.

Additionally, we have experimentally demonstrated that RIP, OSPF, and EIGRP can be customized to provide the desired level of performance by adjusting the appropriate parameters. Specifically, the EIGRP protocol was shown to exhibit the fastest convergence rates in relation to its hello interval. Generalizations such as "a slow convergence rate" are often encountered regarding some of these protocols. We have shown, however, that the convergence rates can be adjusted by the user. Trade-offs, such as increased processor time and additional network traffic, however, must also be carefully considered during deployment. Due to the variety and possible complexity of today's networks, a routing protocol must be chosen based on a number of factors, only one of which is the convergence rate. Although our tests were performed within a hybrid star-ring topology, the principles presented here can be applied to any high-speed network based on an arbitrary 2-connected topology to achieve similar levels of performance from all three protocols.

References

[1] Entrepreneur.com. "Popularity of Public Ethernet Rising." [Online] Available: http://www.entrepreneur.com/article/0,4621,299884,00.html

[2] S. Song, J. Huang, P. Kappler, R. Freimark, and T. Kozlik, "Fault-Tolerant Ethernet Middleware for IP-Based Process Control Networks," *Proc. 25th Annual IEEE Conference on Local Computer Networks*, Tampa, Florida, USA, 2000, 116-125.

[3] J. Huang, S. Song, L. Li, P. Kappler, R. Freimark, J. Gustin, and T. Kozlik, "An Open Solution to Fault-Tolerant Ethernet: Design, Prototyping, and Evaluation," *Proc. IEEE International Performance, Computing, and Communications Conference*, Phoenix/Scottsdale, Arizona, USA, 1999, 461-468.

[4] Linux-HA Development Team. High Availability Linux Project. [Online] Available: http://www.linux-ha.org

[5] W. Zaumen and J. Garcia-Luna-Aceves, "Steady-state Response of Shortest-path Routing Algorithms," in *Proc. IPCCC*, 1992, pp. 323-332.

[6] Y. Zhao, X. Yin, B. Han, and J. Wu, "Online Test System Applied in Routing Protocol Test," in *Proc. Ninth International Symposium on Analysis and Simulation of Computer and Telecommunication Systems*, 2001, pp. 331-338.

[7] R. Jonkman. NetSpec. [Online] Available: http://www.ittc.ku.edu/netspec/

[8] V. Jacobson, C. Leres, and S. McCanne. TCPDUMP Public Repository: libpcap-0.6.2. [Online]. Available: http://www.tcpdump.org/release/libpcap-0.6.2.tar.gz

[9] Internet Engineering Task Force (IETF) and the Internet Engineering Steering Group (IESG). RFC 3220. [Online] Available: ftp://ftp.rfc-editor.org/in-notes/rfc3220.txt

[10] V. Jacobson, C. Leres, and S. McCanne. TCPDUMP Public Repository: tcpdump-3.6.2. [Online]. Available: http://www.tcpdump.org/release/tcpdump-3.6.2.tar.gz

[11] Internet Engineering Taskforce and Internet Engineering Steering Group: Network Working Group. RFC 2988: Computing TCP's Retransmission Timer. [Online] Available: http://www.rfc-editor.org/rfc/rfc2988.txt