FAULT-TOLERANT RECONFIGURABLE ETHERNET-BASED IP NETWORK PROXY

M. Crocker, J. Baranski, G. Lazarou
Department of Electrical and Computer Engineering
Mississippi State University
Mississippi State, MS 39762, USA
{mac1, glaz}@ece.msstate.edu

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 current applications and provides full redundancy with minimal packet loss and fast reconfiguration times. A prototype implementation yielded a reconfiguration time of 1.55 sec. Used in conjunction with optimized dynamic routing protocols, an average recovery time of 3 seconds can be seen. The proposed proxy scheme can be deployed in any network based on a topology that allows two connections between a subnet and the backbone. The solution relies on a dynamic routing protocol to provide backbone level routing around the malfunctioning inter-network link.

Keywords

Networks, Network fault tolerance, Network reliability, Communication system routing, Routing

1. Introduction

Inexpensive and widely available hardware has made Ethernet one of the most widely used link layer protocols in today's IP networks [1]. Fault tolerance, however, still remains a major issue

* Parts of this paper were presented at CIIT'03, © IASTED 2003, and SCI'04, © IIIS 2004.
** This work was partially supported by the US Office of Naval Research (ONR) under Contract Number N00014-02-1-0623.

in IP over Ethernet networks. IP networks built on Ethernet technology can typically be constructed using bus, star, or tree topologies. These topologies inherently allow for uninterrupted network service during the addition, removal, or failure of network nodes. They do not, however, provide a redundant connection to parent networks in a subnetted environment. This lack of fault tolerance prevents Ethernet-based IP networks from being used in situations that require a high-availability network solution.

A number of redundancy schemes for Ethernet networks have been designed and tested over the past few years [2, 3, 4]. In [2] and [3], the authors present a fault-tolerant Ethernet architecture that requires additional software and hardware to be added to each node within the network. Their solution also requires that a specific network architecture be constructed and deployed in order to achieve fault-tolerance. The Linux High Availability project [4] offers a wide range of resources for high availability computing, but these solutions also require additional software or specific configurations to work. Hardware solutions, such as intelligent switches or routers, are an option for providing redundancy, but this requires replacement of existing hardware and the additional functionality significantly increases the cost. We present an approach that focuses on a simple design mated with a specific, but flexible network topology. With this design, we have attempted to create a "plug-in" solution to typical IP over Ethernet networks

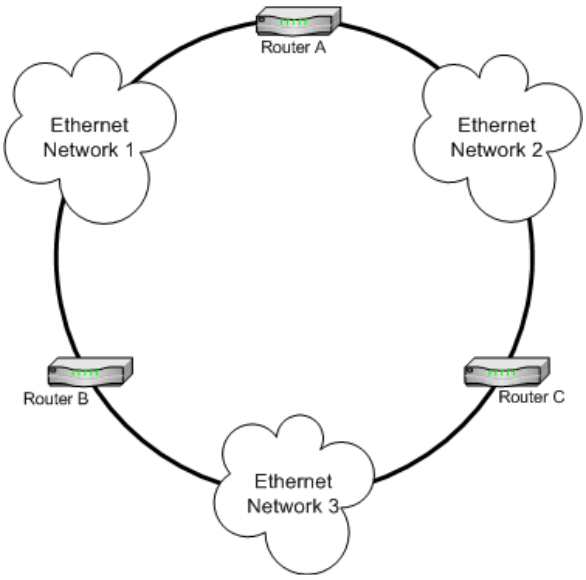

Figure 1 A hybrid star-ring topology

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 accessible to its subnets via a single router. Modern 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 be provided 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 our redundancy scheme, it is one of the most widely used WAN backbone topologies as well as one of the most practical methods of providing two connections to an Ethernet subnet from the backbone. 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 an easily scalable solution. FREPs can be added only to mission-critical networks that must maintain connectivity at all costs, while other networks remain unchanged.

In this paper, we present a general overview of the design of the FREP as well as a detailed description of a prototype FREP implementation. We also evaluate the performance of the FREP and discuss the configuration and resulting performance of three dynamic routing protocols when used with the FREP.

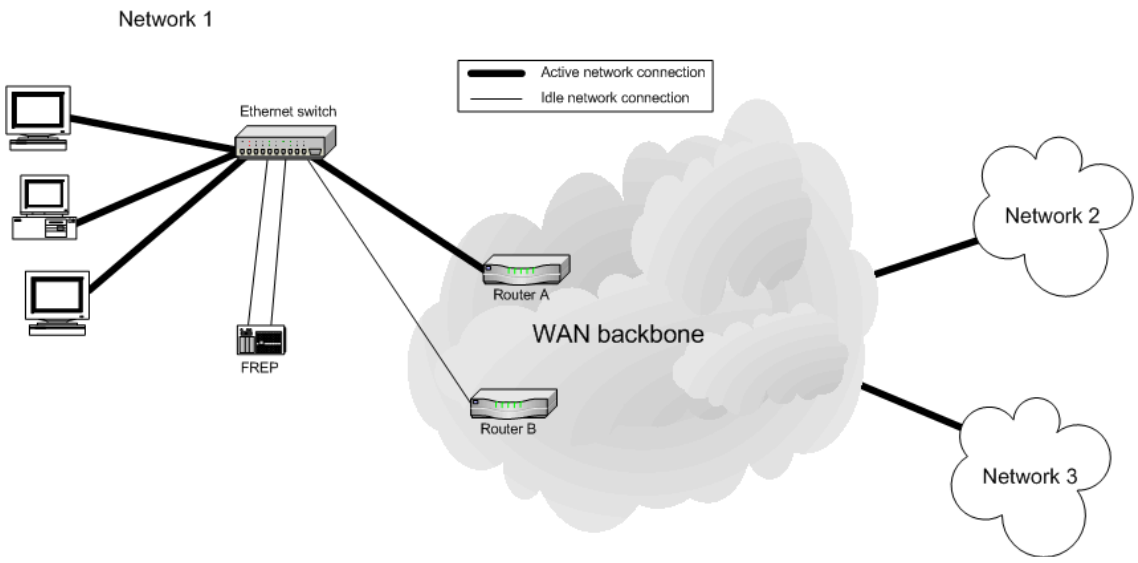

Figure 2 FREP during normal operation monitors primary router for failure

2. Design Overview

The design of the 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 acting as the default gateway for that particular subnet becomes unreachable. The proposed scheme 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.

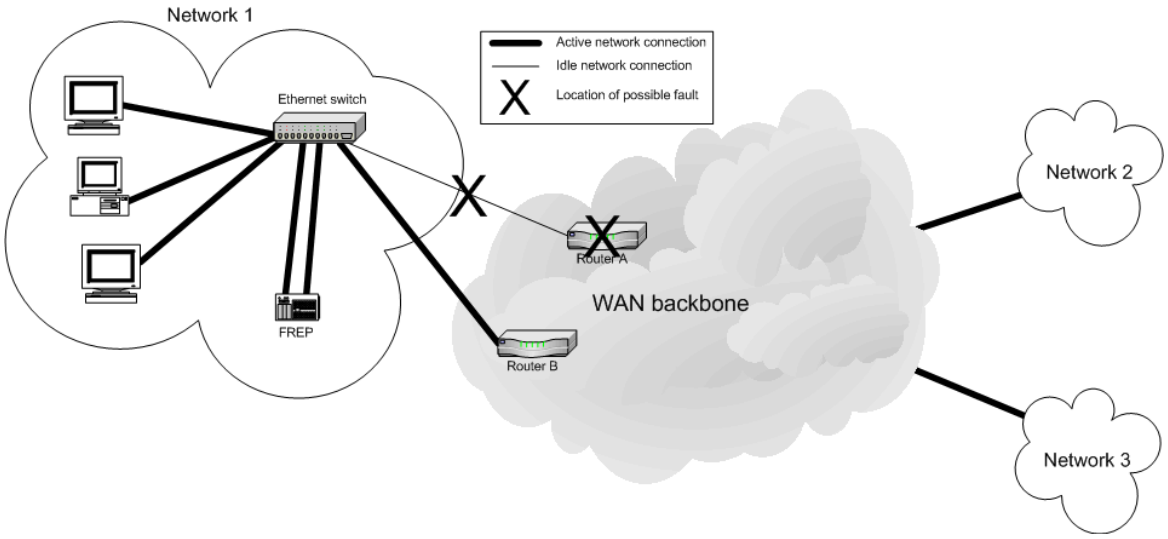

Figure 3 FREP in failover mode, assumes identity of failed router and forwards packets to backup router

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 presence 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 when the link is experiencing intermittent operation or “flapping”.

3. Prototype Implementation

The FREP was implemented using a standard PC with two network interface cards and custom software to perform the failover operations. The software was written in C and is responsible for

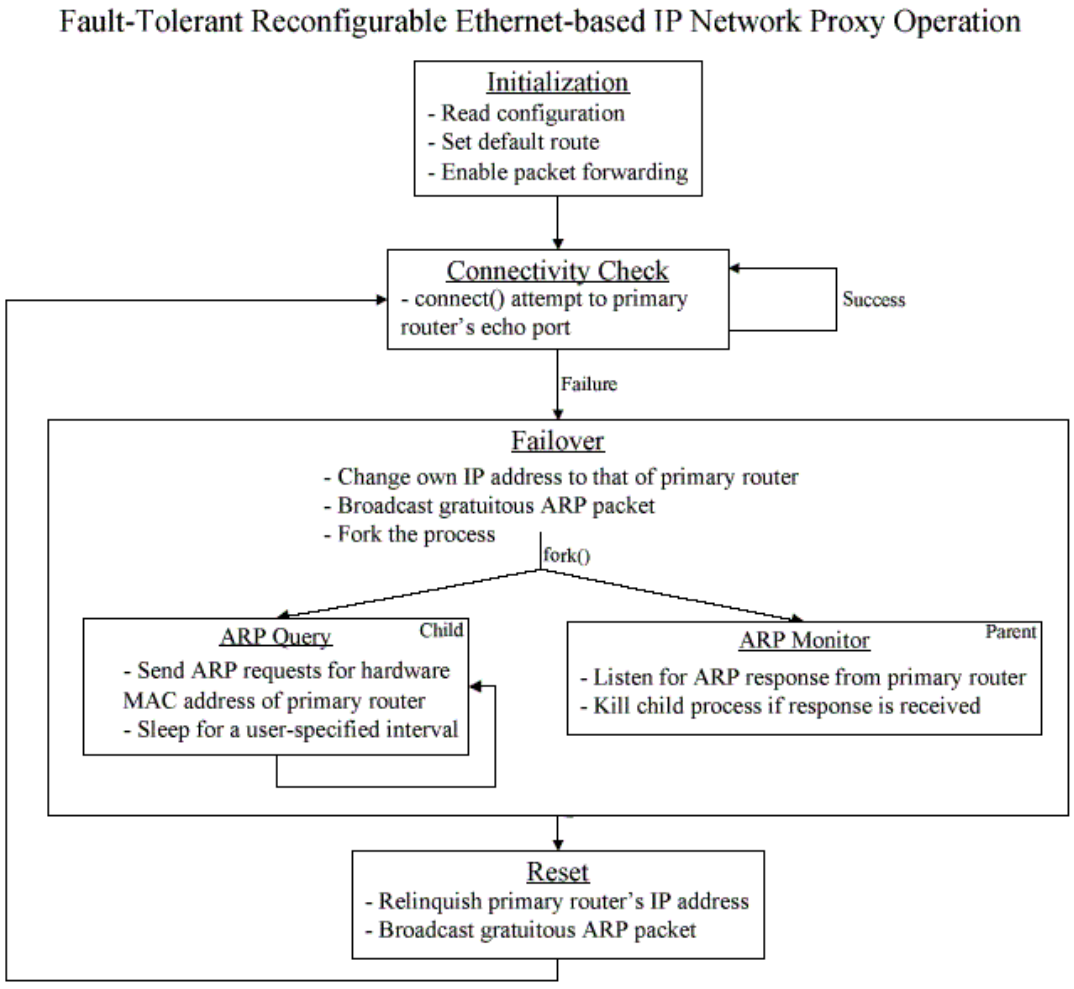

Figure 4 Sequence of operations performed by the FREP

the control of the FREP. It is a user-space program that runs as a daemon. The following information is collected by the software upon initial execution:

- IP address to assign to each network interface (*eth0* and *eth1*)
- The local subnet mask
- IP address and hardware MAC address of the primary router
- IP address of the backup router
- A number of parameter values that affect the frequency of various actions performed by the FREP

Once the values have been properly input, a configuration file is written for future use. The FREP then proceeds to perform 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. All of the operations performed by the FREP are illustrated in Fig. 4.

Testing and implementation of the FREP took place in a test bed network consisting of three subnet/router pairs. We used eight 1.4 GHz Pentium III Dell Poweredge 1650 servers to create three networks with two of the eight servers on each network and the remaining two acting as FREPs. The networks were connected using Linksys Etherfast II switches and three Cisco 1720 routers. The network was designed using a hybrid star-ring topology as described in section 1. We chose a star-ring topology since this is the most commonly used network topology, but our solution will work for any dually connected network. Netspec [5] software was used to place a load on the network during testing to more closely simulate a real-world network.

3.1 Connectivity Checks

The connectivity checks performed by the FREP consist of a connect() system call with a stream socket that points to the echo port[1] of the router. The connect() call is set to time out after a specified interval. The timeout operation is achieved 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. 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

[1] Port 7 is the IANA assigned port number of the echo service [6].

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.

3.2 "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 sending a gratuitous ARP packet[2] to the Ethernet broadcast address. 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 are now being delivered directly to the FREP. Once the packets are received by the FREP, a routing decision is made by the kernel. Since no static routes exist in the routing table, outbound packets are sent via the default route, which was inserted earlier. Thus, without changing the IP address of the router, all packets sent to the primary router on a particular subnet are now forwarded to a backup router via the transparent proxy.

3.3 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

[2] 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. [7]

Linux kernel and are constructed using the arphdr structure and addressed using a sockaddr_ll structure. 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.

4. Performance Analysis

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
- Maximum allowed timeout of TCP connectivity check
- Maximum number of failures allowed 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 [9], 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. Section 5 details the tested routing protocols and how they can be optimized to provide a desirable failover period. 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.

5. Routing Protocol Analysis

We evaluated three dynamic routing protocols, Routing Information Protocol (RIP), Open Shortest Path First (OSPF), and Enhanced Interior Gateway Routing Protocol (EIGRP) used in conjunction with our FREP. Although previous work exists in which routing protocols have been compared ([10], [11]), 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 our FREP, we measured the duration of network outage after a link failure while running one of the aforementioned dynamic routing protocols.

Initial failover tests resulted in convergence times in the area of 40 seconds or more while using the OSPF dynamic routing protocol. The FREP was actively forwarding packets to the backup router after 1.55 seconds but the OSPF routing protocol was configured with the default settings of 40 seconds before a path is considered invalid. Therefore, it took the backup router 40 seconds to acknowledge that the path to the primary router was no longer valid. By changing the various timing settings associated with each routing protocol, we were able to reduce the failover time to an average of 3 seconds.

The design and operation of RIP, OSPF, and EIGRP has been described in detail in various sources ([12], [1], [13]). Their distinct characteristics and advantages/disadvantages are also often cited. However, here we present an approach to the evaluation of the routing protocols based on their performance and ability to reconfigure quickly after a link failure. We tested the protocols in a simple Ethernet-based network and evaluated their performance based on convergence rates and additional network and CPU overhead.

5.1 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. 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.

5.2 OSPF and EIGRP

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 [14]. This behavior forced us to use ICMP-based tests during analysis of the OSPF and EIGRP protocols.

6. Routing Protocol Performance

The duration of network outages while running the RIP, OSPF, and EIGRP protocols was measured using custom designed utilities as described above in 3. Cisco IOS C1700 Software version 12.2(11)T2 was used and, unless otherwise mentioned, the parameters of the routing protocols were left at their default values. The results for analysis of each protocol are outlined individually below.

6.1 RIP

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.

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	2.94
2	6	6	8	5.01
3	9	9	12	10.04
4	12	12	16	14.86
5	15	15	20	18.46

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

6.2 OSPF

Tests of the OSPF protocol were performed using ICMP traffic in the same manner as TCP traffic was used to analyze the RIP protocol. 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		**Outage Duration (sec.)**
Hello interval (sec.)	Dead time (sec.)	
1	3	2.94
1	5	5.01
1	10	10.04
1	15	14.86
5	20	18.46

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.

6.3 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.

Table 3 - Average outage duration of network outage after link failure measured via an ICMP-based analysis of the EIGRP

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

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.

7. Conclusion

7.1 Alternative Methods

Some alternative methods of achieving fault-tolerance via a proxy were also researched. These methods included ICMP redirects, hardware MAC address takeover, and an active proxy scheme. The current method was chosen based on its ease of implementation and efficiency.

7.1.1 ICMP Redirects

Using the method of ICMP redirects is a simple way of redirecting traffic destined for a router. One must ensure, however, that all of the equipment connected to the network is able to accept ICMP redirects and obey them, which may not always be possible. Thus, this technique of forwarding traffic violated the goal of designing a completely transparent redundancy scheme.

7.1.2 MAC Address Takeover

A form of MAC address takeover is implemented in our current scheme. ARP spoofing is used to re-associate IP addresses with different hardware addresses. A similar approach involves changing the MAC addresses of network devices themselves. This approach, however, still requires updates to be made to the ARP tables in devices such as Ethernet switches before any changes will take effect and is difficult to implement.

7.1.3 Active Proxy

Another alternative technique employs an “active” proxy to forward network traffic appropriately. The proxy is placed between the nodes on the local subnet and the two routers for

that subnet. Based on connection status, packets are forwarded to the appropriate router by the proxy. This method, however, forces all traffic to be constantly piped through another device, regardless of connection status. The result is an additional amount of latency that is added to all traffic flowing into and out of the subnet. An advantage of the method, however, is that reconfiguration can take place with very little delay.

7.2 Possible Improvements

As we have measured through our experiments, our FREP can achieve an average failover time of 1.55 seconds, but the failed router is not acknowledged by the dynamic routing protocols until a few seconds later. To solve this problem, the FREP could be modified to notify the backup router as soon as it detects a failure. Obviously, this would introduce much more complexity into the system since the backup router would have to be configured to acknowledge the FREP as a valid source for supplying updates about the network. This additional functionality could also introduce security vulnerabilities if a malicious host found a way to assume the identity of the FREP. We chose not to explore this possible improvement since the slight increase in performance does not justify the increased complexity and risks.

7.3 Packaging

Our prototype implementation of the FREP was based on a standard PC. Production quality applications of this device, however, could be scaled down to a much smaller size. The FREP could, for example, be based on a small form factor PC or other small computer appliance and be specifically tailored for deployment in network closets and server rooms. Furthermore, as layer 3 network switches become more popular, additional functionality is readily being built into the devices. One such addition could be the implementation of a FREP directly inside of an Ethernet

switch. This type of switch would be able to provide redundancy with even faster reconfiguration rates than the current FREP implementation by directly routing packets around a failed link.

7.4 Summary

The described proxy scheme provides an inexpensive and simple way to integrate redundancy into an existing IP over Ethernet network. It is not, however, an ideal solution for true high-availability networks. In these types of networks, more specific solutions such as [4] are often employed. Our proposed FREP scheme provides a transparent method of allowing nodes on a subnet to communicate with a backup router in the event that connectivity with the primary router is lost. The failover operation was shown to have an average duration of 1.55 sec for the FREP along with an average of 3 sec for the selected dynamic routing protocol to acknowledge the failure. The duration of the recovery operation was found to be independent of network topology or routing protocol. Using our scheme, additional redundancy can be provided to an Ethernet network without modification of existing networking equipment or software.

References

[1] L. L. Petersen and B. S. Davie, *Computer Networks: A Systems Approach, Second Edition*. (San Francisco, CA: Morgan Kaufman Publishers, 2000).

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

[6] Internet Assigned Numbers Authority. Port Numbers. [Online] Available: http://www.iana.org/assignments/port-numbers

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

[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] 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

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

[11] 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.

[12] R. Malhotra, *IP Routing*. Sebastopol, CA: O'Reilly & Associates, pp. 10-156.

[13] A. Leon-Garcia and I. Widjaja, *Communication Networks: Fundamental Concepts and Key Architectures*. Boston, MA: McGraw Hill, pp. 590-618.

[14] 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