Creating Redundant Linux Servers

Horms (Simon Horman)
horms@zip.com.au.

To be presented at
The 4th Annual Linux Expo
The Bryan University Center,
Duke University,
Durham,
North Carolina, USA
Thursday 28th - Saturday 30th May 1998

http://linuxexpo.zip.com.au./

I would like to acknowledge the assistance of my employer Zip Internet Professionals `http://www.zip.com.au./` for their assistance and patience that enabled this presentation to come together.

Additionally, I would like to thank Mr. O'Brien, Gus, Miss Kim, K and Raster for their help along the way.

Abstract

For an organisation of any size fault tolerance is an important issue. A server going down should not leave users twiddling their thumbs. A simple solution to this is to create backup servers that can be switched in when a server goes down. Using Linux this can be easily achieved using either existing servers or dedicated backup servers.

Many services have good redundancy built in. Examples of this include mail servers and name servers. However services such as POP3 and manual proxies which require end users to specify a host to connect to, are not afforded such fault tolerance. It is for services such as this that providing backup servers becomes crucial.

The idea is to create a backup server that when called upon assumes the identity of the failed server in addition to any existing identities. The backup server is given an IP alias for the failed host and uses ARP spoofing to convince the rest of the network that the backup server is in fact the failed server.

This method of creating backup servers can be supplemented by using a TCP/IP Switch that allows content based services such as POP3 to be sourced from servers that may have other inaccessible services on them.

Additionally housing the content for services such as HTTP on a dedicated NFS server enables a backup HTTP server to serve a site as well as the primary server.

These are clearly a quick and dirty solutions to creating backup servers. They have however proved to be quite successful in practice and requires little or no outlay for additional hardware.

Contents

1 Introduction

Working for an ISP with Linux servers it became apparent that the built in redundancy in many key services was either inadequate or non-existent. Of particular concern was redundancy in proxy servers. As bandwidth in Australia is relatively expensive mandatory proxies for HTTP are imposed by many ISPs. Manual proxies and the issuing of automatic proxy configuration files are particularly lacking in redundancy. To make this redundant a method of backing up HTTP and proxy servers was investigated. What was required was a generic method for a backup server to take over the role of a lame server.

The idea initially proposed was to update DNS records as required. This would change the IP address of the lame server to that of the backup server. This was found to be unsatisfactory on the following counts;

1. The time to live on the zone files would need to be turned down severely to account for any users using DNS servers other than the master or secondary that can easily be reset for the zone in which the servers lie.
2. Users may access servers using an IP address rather than a host name.
3. Users may use non-DNS methods such as an /etc/hosts file to map server host names to IP addresses.

After some investigation it was found that a solution where the backup server would assume the IP address of the lame server would be ideal. This eliminated the difficulties related to the DNS based solution. The only remaining difficulty was to convince other boxen on the LAN of the change in circumstance and this is where ARP Spoofing came into the game [YV].

ARP spoofing is a method often employed by hackers to assume the identity of a host on a LAN. For this application ARP spoofing allows the backup server to take of the IP address of the lame server.

2 ARP Spoofing

2.1 Background

To implement a redundant server in Linux using ARP spoofing is a relatively simple task. The existing server is given a second interface such that the server can still be accessed when the backup server is in operation. This is best achieved using a second physical interface as this gives better hardware redundancy [HM98]. However in most situations using IP aliasing is quite satisfactory.

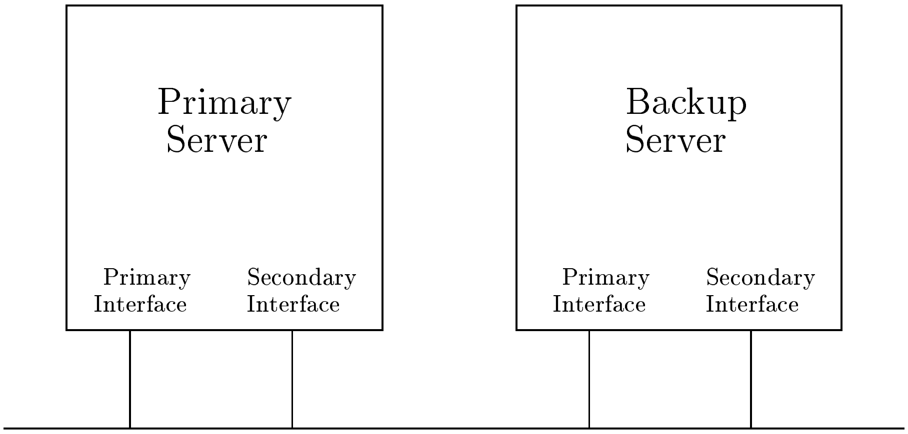

Figure 1: Original and Backup Server Interfaces

2.2 Activation

When the backup server is brought into operation it sets up an interface with the IP address of the server it is to back up. Again this can be an additional physical interface or an IP alias. The backup server then uses ARP spoofing for the duration of its operation to ensure that it receives all packets directed to the server it is backing up.

The spoofed ARP packets that are sent announce the hardware address of the backup server that has an interface for the now lame server's IP address.

These ARP packets are addressed to the broadcast hardware addresses. This is known as a Gratuitous ARP as a machine makes an ARP request for its own IP address.

ARP is central to the functioning of a LAN as it enables the hardware address of a machine to be found given its IP address. Once the hardware address of a machine is know packets can be sent to it over the LAN. Machines keep a cache of hardware to IP address mappings so that a fresh ARP request doesn't need to be sent out for each IP packet. The hardware address in the most recent ARP reply for a given IP address will be used. Hence by using Gratuitous ARP it is possible to force this cache to be flushed, redirecting IP packets to a different hardware address and hence in this case a different machine.

It is important that the ARP packets are sent frequently enough that the ARP cache of other boxen on the LAN does not expire. If the ARP cache did expire then an ARP request for the hardware address of the lame server would be issued. If the lame server is in a state where it is able to answer ARP requests then a race condition would be created between the lame server and the backup server, as shown in Figure 2.

2.3 Deactivation

Once the existing server is ready to be used again it is simply a matter of removing the additional interface on the backup server and stopping ARP spoofing. Finally additional spoofed ARP packets are sent out pointing the existing servers IP address back to the original hardware address.

2.4 Automation

The process of turning on and off the backup server is easily automated such that if the existing server fails the backup server is activated. Such automation takes two stages. Firstly the status of the service is gauged by attempting to access key services it provides. Secondly in a failure situation scripts to enable the second interface on the backup server and kick of ARP

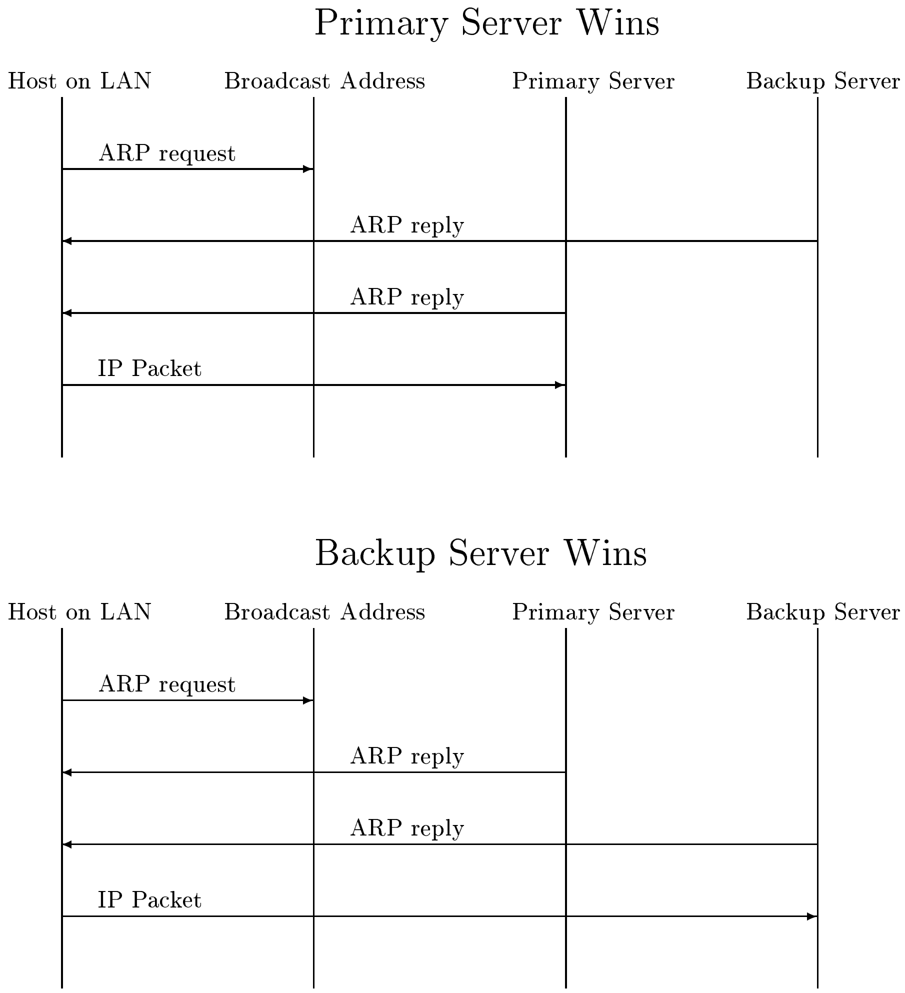

Figure 2: Race Condition for ARP replies

Spoofing are activated. Similarly by accessing the lame server via the second interface it can be ascertained when the backup server can be deactivated by running scripts that deactivate the second interface on the backup server and stopping ARP Spoofing.

2.5 Improvements

The ARP based solution is particularly well suited to services which act as a relay. Proxies and SMTP relays fall into this category and the users should not be able to tell when the backup server is in operation. With this in mind other complimentary methods of creating redundant servers have been investigated. The use of some sort of TCP/IP switch on servers, backup or otherwise, would allow a more powerful backup scheme to be developed as content could still be sourced from servers where it is still available.

3 TCP/IP Switch

3.1 HTTP Accelerator

The popular Squid proxy daemon comes with a facility that allows a single server to act as a front end to web servers [OP97]. This works by having clients connect to the Squid server as if it were an HTTP server and then farming requests off to the real web server or servers. This can be used to share load around multiple servers on high volume sites, as illustrated in Figure 3 or to protect HTTP servers that contain sensitive data by placing them behind a firewall such that the Squid server can access the HTTP server but other hosts on the Internet can not.

Though primarily intended to allow load sharing on high volume sites this can also be used to provide some form of redundancy. The HTTP accelerator server can be a front end for multiple back end http servers hence the loss of a HTTP server should not result in a site being down. And of course as the http accelerator itself has no content is can be backed up using the ARP base method of creating redundant servers. On small sites this extra layer between users and the web server may just be another potential point of failure however the switching idea presented is an interesting one.

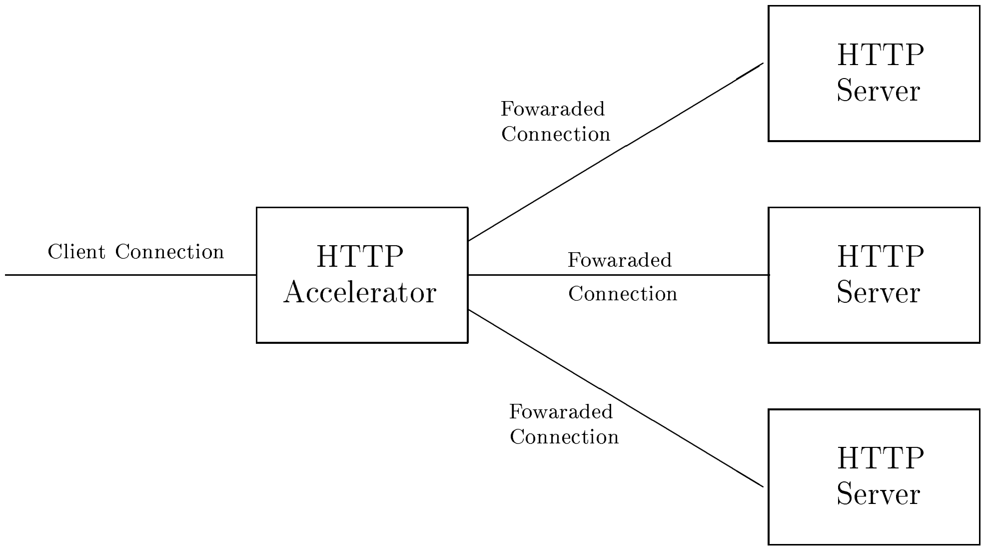

Figure 3: HTTP Accelerator

3.2 POP3 Switch

It is quite common for the SMTP and POP3 servers to be the same box so mail is delivered and collected from a spool directory controlled by a single localised system. In a situation where the SMTP daemon is incapacitated it is desirable to switch to the backup server so users can still send mail. Even if the POP3 daemon was still operable by switching to a backup server, that invariably does not have access to the mail spool and so POP3 also becomes unavailable. However a POP3 Switch can overcome this.

A POP3 Switch is simply a data pipe that accepts a list of foreign host/port pairs and tries them in turn until a connection can be made as shown in Figure 4. So in our situation the POP3 Switch may first try to contact the POP3 port of the lame host and then go to a dummy POP3 server listening on a port on the local host.

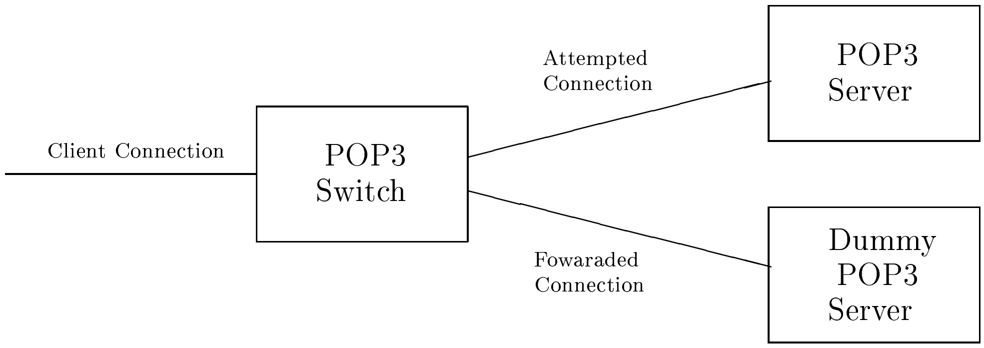

Figure 4: POP3 Switch

3.3 A Generic Switch

Of course the POP3 switch described is just a TCP/IP data pipe and hence is extensible to just about any protocol that uses TCP/IP. The only penalty is that the further down the list of possible host/port pairs the switch has to go before making a connection the longer the connection time becomes. However some sort of caching mechanism by which a bad host/port is not tried again for a time could improve this.

Hence we are able to swap in backup servers using ARP spoofing and have them point to content where it is still available using TCP/IP data pipes.

4 NFS Backbone

So far a method for switching backup servers in to assume the IP address of a lame server has been found and a way to source services from otherwise lame servers has been explored. However if we are trying to back up a service that provides a large amount of relatively dynamic data and the service goes down we still do not have an adequate solution.

An example of such a service is a HTTP server. It is not necessarily practical to keep multiple copies of a web site on different hosts due to the dynamic nature of most sites and the cost in terms of disk space. A solution that enables a backup server to access the content of a service such as HTTP when the main server goes down is to have the content situated on a third server and mounted via NFS.

If the NFS server is set up such that it does nothing but serve NFS it should be quite stable and a low risk single point of failure. Additionally, by placing the NFS server on a physically separate network or on a different segment of the LAN and giving servers that use it a second network card there is no issue relating to extra data on the network.

Therefore the content for the service can be accessed regardless of whether the main server or the backup server is in operation. In the case of an HTTP server for, which this solution is particularly well suited, this means the web site should remain accessible.

5 Choosing a Backup Box

Although all of the solutions discussed do not require a dedicated backup server it is advisable to have one. If a server that has other tasks to perform is run as a backup server then the additional load placed on the server when it is running the services of another box may cause an unacceptable slow down or raise reliability issues. For this reason it is advisable to have a backup server on which very little is running.

6 Testing

As with any system is is important to test that the backup server functions as expected. Your testing regime should include a full production test including having any automated aspects run their due course. Although this will result in some disruption of service to users it is better for a brief outage to occur under controlled circumstances than for some unexpected behavior to surface in a crisis situation.

It is also a good plan to have a regular testing procedure in place. The nature of the backup server is that it hardly ever gets used and is likely to be used for other purposes from time to time. As such it is very easy for one configuration or another to get altered and go unnoticed. By conducting regular, possibly automated, tests you can ensure that the backup server is always in good shape.

7 Discussion

The ARP based solution is particularly well suited to services which act as a relay. Proxies and SMTP relays fall into this category and the users should not be able to tell when the backup server is in operation.

When the service that is to be backed up is a source of data this method of creating redundant servers, though not well suited, can still be successfully applied. A backup POP3 or IMAP server could be configured such that an email explaining the current situation is delivered. Key parts of a web site can be duplicated and warning pages issued in lieu of unavailable pages.

When the ARP based solution is coupled with a TCP/IP switch then services that provide content can also be made more redundant. Finally by housing content on a NFS server backup servers can have access to content and serve it accordingly.

The redundant servers created can be used in a variety of situations. First and foremost their activation can be automated such that the backup servers are called into service in emergency situations. Automation is particularly attractive here as such situations typically occur around 2am. Additionally, the redundancy can be used to prevent disruption to users when system maintenance and hardware upgrades are being undertaken.

We can see that using simple utilities coupled with the power of Linux redundant servers are easy to realise even for small organisation. This redundancy can be used to provide a more constant and stable level of service to users. This increases their satisfaction while reducing your support burden.

While it is obvious that the solutions presented are targeted towards low end applications there is no reason why these concepts could not be scaled up.

What is important to realise is that the power of Linux enables us to create solutions that suit our needs rather than modifying our needs to fit with the solutions available.

8 Glossary

ARP: Address Resolution Protocol. Protocol used to map an interface's IP address to the hardware address of the network card.

Daemon: A programme that runs in the background and performs a specific task. A Web server is usually implemented as a daemon.

Data Pipe: Daemon that accepts a TCP/IP connection from, and forwards it to, another host and port. Note that the host can be any host including the host on which the daemon is running.

DNS: Domain Name Service. Distributed database used to map host names to IP addresses and vice-versa.

Hardware Address: Unique number associated with each network card used with low level protocols.

Host: A computer on the Internet.

Localhost: Interface on a computer that loops back to the computer on which the interface resides on.

HTTP: HyperText Transfer Protocol. Protocol used by the World Wide Web.

IMAP: Internet Message Access Protocol. Protocol used to view mail in remote mail boxes.

Interface: Software access point to network hardware.

IP: Internet Protocol. The underlying protocol used to transfer data on the Internet.

IP Address: Unique number assigned to each interface on the Internet.

IP Aliasing: Kernel option that allows multiple interfaces to be assigned to a single network card.

ISP: Internet Services Provider. An organisation that provides internet connectivity and other related services.

LAN: Local Area Network. Network used to connect boxen at close proximity.

NFS: Network File System: Method of making a directory and its contents available to other boxen on a network.

Redundancy: The ability to keep functioning at some level after a failure.

POP3: Post Office Protocol 3. Protocol used to download mail from remote mail boxes.

Proxy: A service by which requests for information from services such as HTTP are done on behalf of clients and the information is returned to the client. The information collected on behalf of the client, may be kept in a local cache, on the proxy server.

Port: A software access point to a host. Hosts have multiple ports and daemons typically listen on a specific port or ports for connections from clients.

Service: A source of information that users access. e.g. A HTTP server provides web pages.

SMTP: Simple Mail Transfer Protocol. Protocol used to transfer email over the internet.

SMTP relay: SMTP server that forwards email from one box to another.

TCP/IP: Transmission Control Protocol/Internet Protocol. Pair of protocols that provide a connection based service used on the internet for protocols such as HTTP and SMTP.

/etc/hosts: A file on Unix systems that maps host name to IP addresses

References

[HM98] hm@seneca.muc.de Harald Milz. Linux high availability howto. http://www.muc.de./~hm/linux/HA/High-Availability-HOWTO.html, http://sunsite.unc.edu./pub/Linux/ALPHA/linux-ha/High-Availability-HOWTO.html, February 1998.

[OP97] oskar@is.co.za Oskar Pearson. Squid users guide. http://cache.is.co.za./squid/, September 1997.

[YV] jmcdonal@unf.edu Yuri Volobuev. Playing redir games with arp and icmp. http://www.rootshell.com./.