

Complete Linux Server Cluster Solution

By

Garrett Steioff

Submitted to
the Faculty of the Information Engineering Technology Program
in Partial Fulfillment of the Requirements for
the Degree of Bachelor of Science
in Information Engineering Technology

University of Cincinnati
College of Applied Science

June 2005

Complete Linux Server Cluster Solution Draft Document

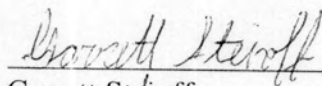
by

Garrett Steioff

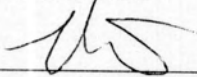
Submitted to
the Faculty of the Information Engineering Technology Program
in Partial Fulfillment of the Requirements
for
the Degree of Bachelor of Science
in Information Engineering Technology

© Copyright 2005 Garrett Steioff

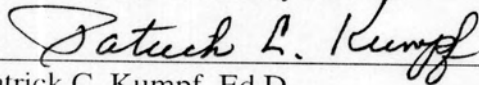
The author grants to the Information Engineering Technology Program permission to reproduce and distribute copies of this document in whole or in part.



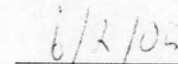
Garrett Steioff



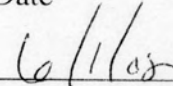
Mark Stockman, Faculty Advisor



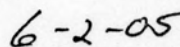
Patrick C. Kumpf, Ed.D.
Interim Department Head



Date



Date



Date

Acknowledgements

I would like to give special thanks to Troy Tolle for allowing me to use hardware, software, and the tech lab at CAS Clifton campus to do my Senior Design Project. I would also like to thank Nick Abt and Brian Sargent for helping me out on a few points when I was having trouble. Finally I would like to thank my parents for giving me support throughout all of this and believing in me.

Table of Contents

Section	Page
Acknowledgements	i
Table of Contents	ii
List of Figures	iii
Abstract	iv
1. Statement of the Problem	1
2. Description of the Solution	1
2.1. User Profile	2
2.2. Design Protocols	3
3. Deliverables	6
4. Design and Development	6
4.1. Budget	7
4.2. Timeline	7
5. Proof of Design	11
5.1. High Availability SLES 9 Server Cluster	11
5.1.1. Heartbeat Failover	11
5.2. LDAP	13
5.2.1. LDAP Replication	14
5.3. Samba	16
5.3.1. CUPS/Samba Printing	17
6. Conclusions and Recommendations	18
6.1. Conclusions	18
6.2. Recommendations	19
Appendix A.	21
A1. ha.cf file	21
A2. haresources file	23
A3. authkeys file	23
A4. slapd.conf file Main Server	23
A5. slapd.conf file Secondary Server	25
A6. smb.conf file	27
References	29

List of Figures

Figure Number	Page
Figure 1. Cluster in Network Diagram	4
Figure 2. Main Server Running	4
Figure 3. Failover to Second Server	4

Abstract

The Complete Linux Server Cluster is a two node server cluster that is designed to be a suitable alternate option to CAS Clifton campus's current Netware 6.5 cluster. Since CAS Clifton campus has gone through many changes in the past few years, they may need change how they support their users. My solution is a cheaper solution that can handle the large user base they have. I am using SUSE Linux Enterprise Server 9 as the operating system for the servers. To handle the users I am using LDAP. For file sharing, I am using Samba to create shares as needed. I have tied CUPS printing into Samba to allow the sharing of printers as well. This would allow CAS Clifton campus to handle their user base in a similar fashion and would cut costs if they needed to.

1. Problem Statement

The University of Cincinnati, College of Applied Science (OCAS) Clifton Campus currently has a Netware 6.5 server cluster connected to a Storage Area Network (SAN) that contains two IBM e325 servers. The cluster is used for authentication into their network, file access and storage, and server side applications. The servers support 14 labs and about 300 faculty and staff. There are about 30,000+ users that have an account on the cluster (6). There are so many students because most of them are from different colleges that are part of the University of Cincinnati but take IT related courses at the OCAS Clifton campus.

The current solution works well for their needs. It can support all the users, it has the storage capacity, but Netware 6.5 licensing is extremely expensive even with academic discounts. Since OCAS Clifton Campus merged with OCAS Victory Parkway, their budgeting has changed dramatically. OCAS Clifton Campus is looking for a networking solution that is very similar to their current setup, but is reduced in the cost. Troy Tolle, the Sr. Systems Engineer at OCAS Clifton Campus, has always wanted to look into alternative solutions to the system they have right now, but has never had the time to look into them (6).

2. Description of the solution

The cluster will have two computers, each has identical hardware. Between the servers will be a SCSI external drive that will connect to both servers. This is to store the mission critical data such as the home directories.

SUSE Linux Enterprise Server 9

The network operating system that will be on each server is SUSE Linux Enterprise Server 9 (SLES 9). The main services that will be used on the servers are Samba 3, LDAP, CUPS, and Heartbeat. Samba 3 will be used mainly for file sharing and storage. Lightweight Directory Application Protocol (LDAP) is used for the directory service on the server, it will contain all the types of the users that will be able to authenticate to the cluster. Common Unix Printing System (CUPS) will be used to handle the printing, it will be tied in conjunction with Samba to make the print job handling less complicated. Finally the Heartbeat service is what controls the fail over of services between the servers. It sends a signal between the servers in the cluster and basically if it doesn't receive a response from a server in a certain time it will be classified as dead. Then the other server will activate the services that were running on the primary server. This should all happen within a few seconds, so no one should even know it happens unless they try to access something just as the server failed.

2.1 Users

Student – A student will have an account on the server so they can log in and will be given storage space to store files for their classes. This account will have only read access to their directory on the server, and nothing else.

Staff/Faculty – They will have access to their home directory and to several shares that are setup for the faculty and staff. They will not have access to the server's main running files.

Technician – They will be given more privileges to server and can access certain configurations to help maintain and tweak settings as needed. They will have their own home directory, and have access to certain technician shares.

Administrator – Total control over the servers. Will have full access to everything on the server and can change anything that needs to be changed.

2.2 Design Protocols

The high availability cluster is designed to keep the critical services up and running so the users have access to network resources if something would happen. For my project I have the basic setup, a two server cluster. For the best setup it is good to have identical hardware for each server. This makes maintaining, upgrading, and setup of the servers a bit easier. Each server has two network cards in them. One is used for Heartbeat software, which manages the fail over. They are connected with an Ethernet crossover cable so they can send a signal back and forth between each other. If the signal does not respond for a period of time on the server that is running your main services, it will be classified unreachable, and the other server will startup the services that need to be run. Figure 1, on the next page, is a basic diagram of the server cluster and few peripherals. The shared hard drive between the servers houses data that needs to be accessed at all times. So users home folders and shared documents will go on that drive. Figure 2 and 3, on the next page, is diagram of the fail over working on the cluster.

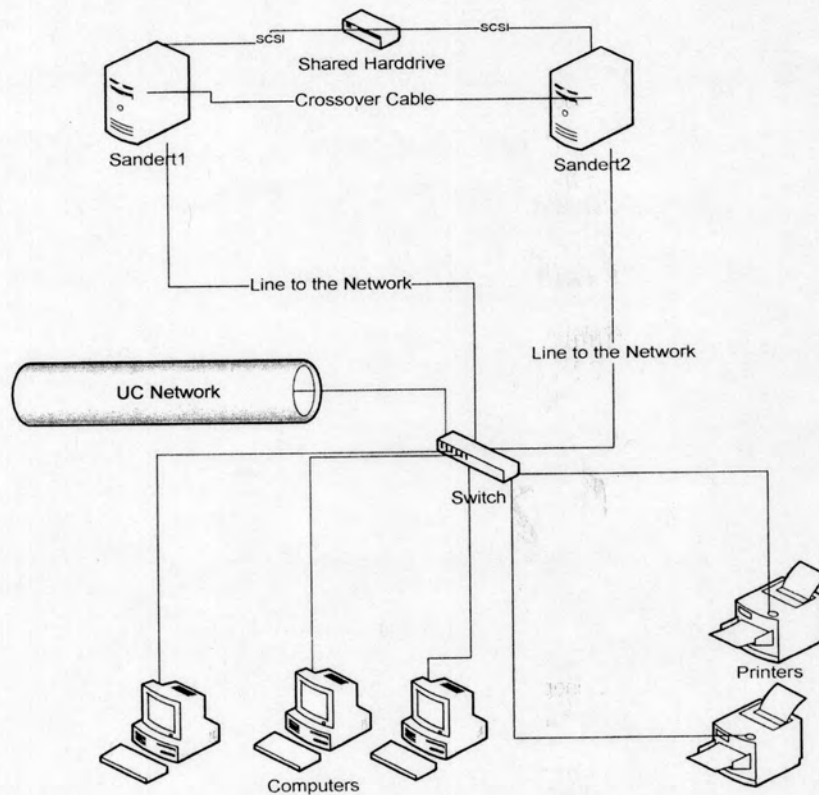


Figure 1.

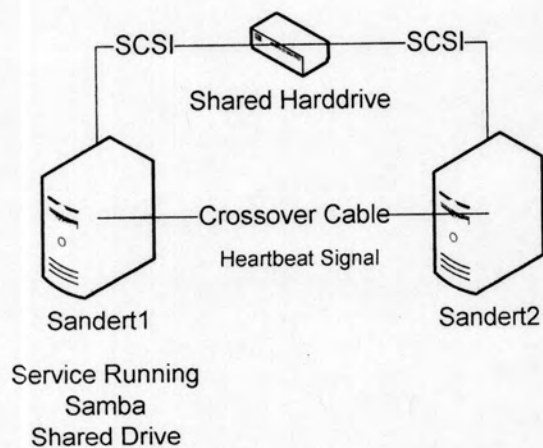


Figure 2.

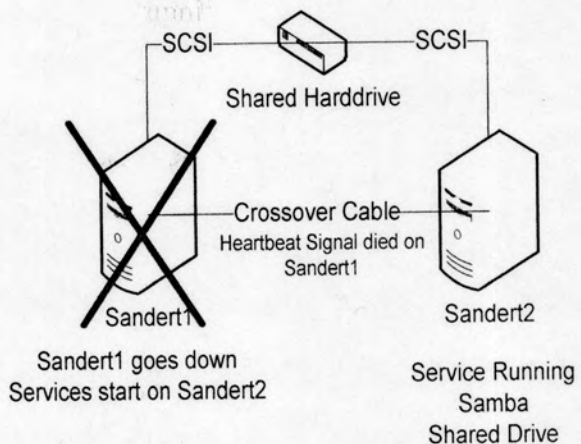


Figure 3.

As the diagram shows if the main services are running on Sandert1, and it goes down, the Heartbeat signal will not be able to continue to transmit between the servers. So Sandert1 will be considered dead and Sandert2 will start the services needed on it.

This setup is for the access of critical data it will not be designed to load balance the services. LDAP will be setup similar to load balancing but will not be run that way.

Critical data will be on the shared drive connected by the SCSI cables. So each server can get access to them. The services though just have to be configured the same, so when one server goes down and the other starts the services up and they run the same way. This only requires that the configuration files match on both servers for each of the services that need to be run.

Each server has a static IP address; they have access to the Internet and the UC network. The Heartbeat is configured to run the services under a shared IP address which is the cluster IP. So when client connects to the cluster they connect to the cluster IP address, not one of the static IP addresses of the servers. Once Heartbeat detects that one of the servers has gone down it releases the cluster IP address and the other server picks it up.

Samba will be used mainly for file sharing, but it will be tied directly with LDAP for authentication, and with CUPS for printing. For the moment I will be working with Windows clients. When someone logs into the cluster they must authenticate with Samba, but LDAP is much more secure and capable of handling a large user base. Since each service has its own username and password database, the problem is linking the two so they are identical. My first thought was scripting to create a user in both Samba and LDAP. After further investigating I found that when creating a LDAP user in YaST I could configure the options for the Samba account as well. YaST stands for Yet another Simple Tool, and it can be related to what the control panel is in Windows. You can install/uninstall packages, configure options for hardware/software, and change simple

options like the date and time. Then after a quick test I found that the LDAP user could log into Samba shares with that account. So it could handle the LDAP options as well as the Samba options. I planned on mimicking CAS Clifton campus's Edirectory setup with LDAP. Edirectory is Novell's directory service for handling user and computer objects, just like Windows Active Directory handles its user and computer objects.

Printing with CUPS linked to Samba helps make printer access much simpler. I will be able to give certain users rights to certain printers. Samba only needs a few simple configuration settings to allow it use CUPS to handle printing. Then with CUPS directly I can add network printers, handle driver settings, and manage print queues. This will handle all the network printers, not any of the local printers that some faculty members have directly connection to their computers.

3. Deliverables

To keep the project reasonable for the time frame, these are the objectives I felt that needed to be done. I defined these deliverables during the design phase of the project.

1. A two node high availability server cluster with failover using SLES 9 operating system and other open source tools
2. File Sharing/Home Directories with Samba for users that log into the cluster
3. Authentication using LDAP for users that log into the cluster (try to make it an only one login for all resources)
4. Printer support with CUPS

4. Design and Development

4.1 Budget

Item	Cost	My Cost
SUSE Linux Enterprise Server 9	\$832.00	\$0.00
Old 1.0GHz Machine	1,200.00	0.00
New AMD Athlon 3000+	1,300.00	0.00
Two Server Boxes	(827.65 a piece) 1,655.30	0.00
Two SCSI Cards	(34.99 a piece) 69.98	0.00
Two Network Cards	(22.00 a piece) 44.00	0.00
Crossover Cable	5.99	0.00
External SCSI hard drive	135.00	0.00
Total	\$5331.27	\$0.00

4.2 Timeline

- **Senior Design I** – September 22, 2004
 - a. September 23, 2004 – First class, started thinking of ideas
 - b. September 27-29, 2004 – Spoke with Professor Stockman on possible ideas.
 - c. September 29, 2004 – Spoke with Troy Tolle on some possible ideas
 - d. October 2-7, 2004 – Get more information on ideas and start researching some.
 - e. October 14, 2004 – Turn in first progress report
 - f. October 14-21, 2004 – Narrow down idea and research on it some more
 - g. October 21, 2004 – Turn in Problem/Need/Area of Inquiry
 - h. October 24-26, 2004 - Gather some research on topic
 - i. October 28, 2004 – Hear guidelines to proposal and begin to get ideas on writing it
 - j. October 28-November 4, 2004 – Write proposal
 - k. November 1-4, 2004 – Get proposal looked over and make corrections

- l. November 4, 2004 – Turn in rough draft proposal
 - m. November 5-18, 2004 – Clean up rough draft and work on power point presentation
 - n. November 18, 2004 – First group presentations
 - o. November 19-December 1, 2004 – Finish up power point presentation and correct any minor details on proposal
 - p. December 2, 2004 – Second group presentations, which is when I go, also turn in presentation and final proposal
- **Senior Design I End** – December 9, 2004
 - **Senior Design II** – January 3, 2005
 - a. January 3, 2005 – Begin working on project itself
 - b. January 4, 2005 – Put together hardware needed and install SLES9
 - c. January 10, 2005 – SLES9 installed, unforeseen hardware problems gave me trouble
 - d. January 11, 2005 – Start installing and configuration of Heartbeat
 - e. January 13, 2005 – Found I needed to add an extra network card to each server and make a crossover cable to make heartbeat work properly
 - f. January 14, 2005 – Made a crossover cable in the networking lab
Professor Stockman aloud me to use some materials up there. Installed the cable and completed setting up heartbeat, and tested failover/failback
 - g. January 17, 2005 – Start configuring Samba for file sharing
 - h. January 19-24 – Unable to work directly on project other class school had to be completed first and start working on Document Freeze

- i. January 25 – Found out that Samba and LDAP must be synchronized somehow. Looking into scripting maybe or another solution. Continue working on Document Freeze
- j. January 27 – Found an open source tool that helps create a Samba and LDAP user simultaneously so their credentials match up, though its configuration looks tricky and Samba and LDAP seem like they have to be configured the way I want them at their final settings. Continue work on Document Freeze
- k. January 29 – February 4, 2005 - Studying for mid terms, minor work on Document Freeze, not much time to continue research on linking configuration between Samba and LDAP
- l. February 5-10, 2005 – Finish draft of Document Freeze to turn into Professor Stockman and Professor Humpert
- m. February 11, 2005 – Continue work on configuration settings to setup Samba and LDAP properly. As of now though project has some functionality to demo
- n. February 14-24, 2005 – Get the configuration setup correctly and have Samba and LDAP work together so users will have a single logon to the cluster.
- o. March 10, 2005 – Senior Design II presentation, work on any corrections or additions to the Document Freeze and continue work on project
- **Senior Design II End** – March 20, 2005
- **Senior Design III** – March 28, 2005

- a. March, 28 2005 – Classes begin continue work on project
- b. April 1-8, 2005 – Work on proper configuration of printing
- c. April 11-15, 2005 – Printing is working looking into proper setup of the shared drive
- d. April 18-22, 2005 – Shared drive not working properly looking into another method of shared storage
- e. April 25-29, 2005 – Hit a dead end on the on the other method of shared storage. Worked on LDAP replication and having problems getting it to work. Begin work on final report
- f. May 2-6, 2005 – Took a few days off thinking that maybe coming back to the project after a break I might see some errors why LDAP isn't replicating. Came back to find I made a small error in the configuration for replication, now it is working properly. Continue work on final report
- g. May 9-13, 2005 – Found how to get my original shared drive working properly and LDAP is working with Samba. Continue work on final report
- h. May 16-19, 2005 – Thought LDAP and Samba were working together properly, found out through more testing they were not. Was able to configure Samba properly to use LDAP authentication. It is working correctly now for Tech Expo
- i. May 20, 2005 – Tech Expo
- j. May 23-27, 2005 – Work on final report documentation
- k. May 26, 2005 – Give final presentation

l. May 30-31, June 1-3, 2005 – Finish up final report, get the final OK on it, and make corrections as needed.

m. June 8, 2005 – Final Bound Report due

- **Senior Design III End** – June 10, 2005

5. Proof of Design

I am creating a high availability cluster which is comprised of two computers that appear to be one server to the users that log in. It is used to keep users' mission critical data available so that if one computer crashes the second can take over the operations that the first was running.

5.1 High Availability SLES 9 Server Cluster

The first of my deliverables is creating the server cluster that is designed for high availability with failover. The operating system for the servers is SUSE Linux Enterprise Server 9. Its installation is similar to any other operating system. I put the installation CD in and went from there. It prompted me with options to configure hardware, and install extra applications. For most of the options I went with default. For the network card options, I set the static IP addresses for each of the cards in the machine. I was also prompted about additional applications to be installed and I made sure LDAP and Heartbeat were checked to be installed. I did this installation on both servers. Once they were both up and working I had to setup failover to make them a high availability cluster.

5.1.1 Heartbeat Failover

Failover was achieved using the Heartbeat software that is set of open source scripts for download off the Internet. To be properly setup Heartbeat uses three configuration files, the *haresources* file, *authkeys*, and the *ha.conf* file (2). These files go

in the folder to where Heartbeat was installed my default location was `/etc/ha.d/`, so that is where I placed the files. The computers must also be able to communicate with each other through the network they are on or through direct connection like a serial or crossover cable. I have a second network card in each with a crossover cable connecting them for the Heartbeat signal. The *haresources* file must be identical on every node in the cluster. It contains the shared IP address that is switched to which ever is the active node. It also tells which server will be the main node and which services will need to be activated on failover. The *authkeys* file is used to setup the authentication Heartbeat uses to send between each of the nodes. The available methods for authentication are `crc`, `sha1`, and `md5`. `Sha1` is believed to be the best and that is what I am used (2). Those are the types of encryption for sending the heartbeat signal packets between the servers. I could not find any more information why `Sha1` is said to be best. The *ha.conf* file is used for many different settings with Heartbeat to fine tune its workings. The important options that need to be set are the node names in the cluster, and the medium type that Heartbeat will be broadcast over. Most of the other options for a basic setup can be left to default. In my cluster the two nodes are `sandert1` and `sandert2`. I am using `bcast` which is for Ethernet, and then I specify that I want it to use `eth1` device. The `eth1` is the second network card I have in each server that is connected with a crossover cable. This is so the Heartbeat signal can be sent over a direct line, and is not affected by the outside network. Once all three configuration files were made and in the proper location on both servers Heartbeat can be started. This is done either by starting it in YaST, or running the Heartbeat startup script from the command line. To run it from command line I just type

“heartbeat start” in the directory where the script is. By default scripts are stored in the `/etc/init.d/` folder.

Once Heartbeat was running on both servers I tested it to make sure failover was working. I setup a simple Samba share for this test and put Samba as one of the applications I wanted to failover in the *haresources* file. Then I turned off Heartbeat on the first server where Samba was running and watched it turn off Samba on that server then it started it over on the second server.

5.2 LDAP

To store user's credentials I used OpenLDAP, which is an open source implementation of LDAP. On installation of the SUSE operating system, I chose the option to install and configure LDAP. The configuration on installation was just a few basic options like setting the password for the administrator account for LDAP. SUSE handled pretty much all of the initial configuration in the *slapd.conf*. For the complete walkthrough on how to compile and install OpenLDAP from start to finish, visit www.openldap.org. They have the latest versions of OpenLDAP for download, and all the information needed to configure it.

The *slapd.conf* is the main configuration file for LDAP. One of the first options that I set was to include the schema file for Samba. The scheme files let LDAP know what type of objects can be stored in the database, and what attributes can be set for them. By default in my *slapd.conf* file there were 5 schema files already included. The *core.schema* file must be there because as the name implies, it has the core objects for LDAP (8). To add the schema file you simply add a *include* statement with the path to the schema file following the *include* statement. By default schema files are stored in

/etc/openldap/schema/. If Samba has been installed it should place the *samba.schema* file in that directory (8). Once this is included, LDAP now can store objects and attributes based on Samba user credentials.

5.2.1 LDAP Replication

Since SUSE handled most of the basic configuration, my *slapd.conf* file was pretty much complete. The main aspect I worked on configuring was making sure LDAP would replicate. So if the main server failed all the users credentials would be on the second server and they could continue working as normal. Configuring LDAP to replicate is actually a very simple process. The first step is to make sure the LDAP service is not running on both servers. This is simply done by typing *ldap stop* at the directory where the service runs, default location on SUSE is */etc/init.d/*. Once LDAP is stopped on both servers you can configure their *slapd.conf* files for replication. On the main server you will add these lines to end of your *slapd.conf* (8).

```
relogfile      /var/lib/ldap/repllog
replica host=slave.example.com:389
      binddn="cn=admin,dc=example,dc=org"
      bindmethod=simple
      credentials=secret
```

For the *relogfile*, you can pick any location to store the log files. *Replica host* is where you put the host name or the IP address of the secondary server you want to replicate to. I used the IP address of my secondary server in my configuration. The *binddn* is the administrator, or an account that has read/write privileges to the LDAP database. It is recommended that you create a new LDAP user for this purpose for security reasons. I have used the Administrator account on my configuration for simplicity. The *bindmethod* should be set to simple, and the credentials should equal the password of the

user that was listed in the *binddn*. The example shows secret because the administrator's default password on installation is secret.

Now on secondary servers *slapd.conf* file the following lines need to be added (8).

```
updatedn      "cn=admin,dc=example,dc=org"  
updateref     ldap://master.example.org
```

The *updatedn* should be the same account that was used on main server's configuration under *binddn*. Finally the *updateref* should be the host name or the IP address of the main server. Like before I used the IP address of the server and not the host name.

The next step is to copy the main servers LDAP database to the secondary server so that they are identical. By default the LDAP database is stored in */var/lib/ldap/*, so you can essentially copy all the database files on the main server to same location on the second server and overwrite them. The other way to copy the database, also the recommended way, is to use the command *slapcat* to output your database into an *ldif* file. *Ldif* files are the clear text files that can contain your whole database or single users. These files are used to add users to your database through the command line. To send an output of your whole database into a file simply type *slapcat > /location/filename.ldif*. Copying the database this way avoids any incompatibilities if you have two different servers configured differently. Once you have the *ldif* file take it to the second server, delete any database files in */var/lib/ldap/* folder then from the command line type *slapadd < /location/filename.ldif*. This will ensure that both databases are identical. Now that *slapd.conf* files are configured, and the databases are identical on both servers, the LDAP service can be started first on the secondary server, then on the main server. Now the *slurpd* service can be started on the main server. To do this it is just like LDAP service by typing */etc/init.d/slurpd start* in the command line. The *slurpd* service checks the

slapd.conf file for the replication options and will automatically replicate any changes to the database that are made. With this configured I added a new LDAP user, and then checked the secondary server and saw that the new user was added to that server as well.

5.3 Samba

Samba is an open source tool used for file and print services that can cross platforms. I use Samba to setup basic file shares, and grant users access to their home directories. I have configured Samba to use the LDAP credentials instead of the default Samba user and password file. For detailed information of compiling and installing Samba visit www.samba.org.

The *smb.conf* is the main configuration file for Samba. The *smb.conf* is clear text so you can edit it with any text editor. Each section in the *smb.conf* that has the brackets around them, for example *[testshare]* are considered to be a share on the server. The *[global]* is special because all settings in that section apply to every share, not just a single one. There are so many settings that can go in the *[global]* section of the configuration file. The *man* pages, which is Linux version of a commands help file has details on all settings. To view the *man page* simply type *man smb.conf* at a command prompt to see all the settings and descriptions on them.

The main settings that I focused on were the ones to link Samba to LDAP. I found that there are two ways to set Samba to use LDAP as its database to store user credentials. The first way is when compiling and installing Samba, there is an option to compile it *with ldapsam*. This would configure Samba through its installation to use LDAP for its user's credentials. It also allows several other LDAP settings to be used in the *smb.conf*. To compile Samba with LDAP settings check www.samba.org, it has

directions on this. The second way to configure Samba to use LDAP is the way I did it and use some of the LDAP settings in the *smb.conf* that are available without compiling and installing it with the *with ldapsam* option. The first option is *passwd backend =* option, this should be set even if you are not using Samba in conjunction with LDAP. The default setting for this option is *smbpasswd*, which means it stores its usernames and passwords in the *smbpasswd* file. Here I changed it to *ldapsam:ldap//servername or IP address*, this tells it to use LDAP to store its credentials and tells it where the database is located. The next setting I put into the configuration was the *ldap admin dn =*. Here I put the administration account for the LDAP database. This is example of the format it should be in, *ldap admin dn = cn=admin,dc=example,dc=org* (Carter 166-169). Samba now needs the option to know the top level of the LDAP database. This option is the *ldap suffix*, and should be formatted like this *ldap suffix = dc=example,dc=org*. Finally the last option to set that is helpful is the password sync. This lets you change the user's passwords through the LDAP utility, or simply use the Samba command line tool. This is its format, *ldap sync = yes*. After Samba is configured I found it was still not authenticating properly. I found that a command had to be run, that command is *smbpasswd -w secret* (Carter 166-169). This command can only be run by the root user because the *secrets.tdb* can only be read or written to by root. This command puts *ldap admin dn* into that file so Samba knows what it is. After that command was run I could authenticate to the cluster and have access to the shares that were needed. To view my configuration file, please refer to Appendix A6.

5.3.1 CUPS/Samba Printing

I used CUPS to manage printing. To make it easier to share, CUPS can work with Samba to share the printers. With the installation of SLES 9 CUPS should be installed by default. To have Samba use CUPS printing first I put the line *printing=cups* in the [global] section of smb.conf file (Ts, Et al 332-333). This lets Samba know to use CUPS for printing. Then I added two printer sections to the smb.conf that need to be added now. They are [printers] and [print\$] sections. The [printers] section options like where to store temporary files and valid users would go here (Ts, Et al 324). The [print\$] is a global share that will affect all the printers. Once these were configured I restarted Samba and CUPS services to make sure the new settings took effect. Now I could go to localhost:631 in the web browser to get the web interface to add printers and other administrator options.

A small problem I found trying to log into the web based tools to add printers, is that the root super user could not log in. Since LDAP is tied into users on the server itself, I had to create a LDAP user and give them access similar to root's access. Once I did this I could log into web based tools as an administrator and manage printers. I then got the IP address from the printer in the tech lab where my servers were located. The options on the web based tools are very simple to follow, like "Manage Printers", and "Add Printers." I then just added the printer that was located in the same room as my servers. I tested the printer from the servers and logged into my cluster with a test user and printed from a lab computer as well. To view my Samba configuration check Appendix A6 to see the printer related sections.

6. Conclusion and Recommendations

6.1 Conclusion

This project was for CAS Clifton Campus as an alternate solution that if they needed, could put it into production. It was designed to follow their business practices and guidelines using SUSE Linux Enterprise Server 9 and other open source tools. I was able to create the basic framework of their current solution by having authentication, file sharing and printing. I used SUSE SLES 9 for the operating system, LDAP for the directory service, Samba for file sharing, and CUPS for printing. I have fulfilled my deliverables as stated in the document freeze.

6.2 Recommendations

This project had a huge learning curve for me. I did not have any experience with Linux or any of the open source tools until I started to work with them for this project. In general I found that most of the tools I used were not too hard to figure out by themselves. It was when I wanted to link things like Samba and LDAP is when became complicated. The main thing I would recommend to anyone working on a similar project or working with these tools is to read as much as you can. Read it very carefully as well, I was stuck on my getting my shared drive to failover and been seen by both servers. I actually read over the solution once before to quickly because I had so many other things to read. There are so many things on the Internet about these subjects, and many different books. Read the *man pages* for all the configuration files, and the tools you are using. I had plenty of information for this, it was just sorting exactly what I needed to use to be my biggest hurdle.

I also recommend just playing around with some of the tools as your trying to configure them. On many occasions I would change just one line in the configuration of Samba and then test to see if changed how it worked. Then I would go back and try

changing that setting another way. At times I found it to be a very tedious exercise, but it helped ensure that it was working the way I wanted it to work.

Appendix A.

Configuration Files

A1. ha.cf file

```
# There are lots of options in this file. All you have to have is a set
# of nodes listed {"node ...} one of {serial, bcast, mcast, or ucast},
# and a value for "auto_failback".
#
# ATTENTION: As the configuration file is read line by line,
#             THE ORDER OF DIRECTIVE MATTERS!
#
# In particular, make sure that the udpport, serial baud rate
# etc. are set before the heartbeat media are defined!
# debug and log file directives go into effect when they
# are encountered.
#
# All will be fine if you keep them ordered as in this example.
#
# File to write debug messages to
debugfile /var/log/ha-debug
#
#
# File to write other messages to
logfile /var/log/ha-log
#
#
# Facility to use for syslog()/logger
logfacility    local0
#
#
# A note on specifying "how long" times below...
#
# The default time unit is seconds
#     10 means ten seconds
#
# You can also specify them in milliseconds
#     1500ms means 1.5 seconds
#
#
# keepalive: how long between heartbeats?
```

```

keepalive 2
#
#      deadtime: how long-to-declare-host-dead?
#
#      If you set this too low you will get the problematic
#      split-brain (or cluster partition) problem.
#      See the FAQ for how to use warntime to tune deadtime.
#
deadtime 30
#
#      warntime: how long before issuing "late heartbeat" warning?
#      See the FAQ for how to use warntime to tune deadtime.
#
warntime 10
#
#      Very first dead time (initdead)
#
initdead 120
#
#      What UDP port to use for bcast/ucast communication?
#
udpport      694
#
#      What interfaces to broadcast heartbeats over?
#
bcast eth0      # Linux

#      The default value for auto_failback is "legacy", which
#      will issue a warning at startup. So, make sure you put
#      an auto_failback directive in your ha.cf file.
#      (note: auto_failback can be any boolean or "legacy")
#
auto_failback on
#      Tell what machines are in the cluster
#      node  nodename ... -- must match uname -n
node sandert1
node sandert2
#
#      Less common options...
#
#      Treats 10.10.10.254 as a psuedo-cluster-member
#      Used together with ipfail below...
#
ping 10.78.3.0

```

```
#
#   Treats 10.10.10.254 and 10.10.10.253 as a psuedo-cluster-member
#   called group1. If either 10.10.10.254 or 10.10.10.253 are up
#   then group1 is up
```

A2. haresources

```
sandert1 10.78.3.245 Filesystem::/dev/sda1::/SecondShare::ext3 smb
```

A3. authkeys

```
#   Authentication file. Must be mode 600
#   Must have exactly one auth directive at the front.
#   auth    send authentication using this method-id
#   Then, list the method and key that go with that method-id
#   Available methods: crc sha1, md5. Crc doesn't need/want a key.
#   You normally only have one authentication method-id listed in this file
#   Put more than one to make a smooth transition when changing auth
#   methods and/or keys.
#   sha1 is believed to be the "best", md5 next best.
#   crc adds no security, except from packet corruption.
#       Use only on physically secure networks.
#
auth 1
1 sha1 whateverwordorphraseforakeyhere
```

A4. slapd.conf Main Server

```
include      /etc/openldap/schema/core.schema
include      /etc/openldap/schema/cosine.schema
include      /etc/openldap/schema/inetorgperson.schema
include      /etc/openldap/schema/rfc2307bis.schema
include      /etc/openldap/schema/yast.schema
include      /etc/openldap/schema/samba3.schema
# Define global ACLs to disable default read access.

# Do not enable referrals until AFTER you have a working directory
# service AND an understanding of referrals.
#referral    ldap://root.openldap.org

pidfile      /var/run/slapd/slapd.pid
argsfile     /var/run/slapd/slapd.args

# Load dynamic backend modules:
modulepath   /usr/lib/openldap/modules
# moduleload back_ldap.la
# moduleload back_meta.la
```

```

# moduleload back_monitor.la
# moduleload back_perl.la

# Sample security restrictions
#   Require integrity protection (prevent hijacking)
#   Require 112-bit (3DES or better) encryption for updates
#   Require 63-bit encryption for simple bind
# security ssf=1 update_ssf=112 simple_bind=64

# Sample access control policy:
#   Root DSE: allow anyone to read it
#   Subschema (sub)entry DSE: allow anyone to read it
#   Other DSEs:
#       Allow self write access to user password
#       Allow anonymous users to authenticate
#       Allow read access to everything else
#   Directives needed to implement policy:
access to dn.base=""
    by * read

access to dn.base="cn=Subschema"
    by * read

access to attr=userPassword,userPKCS12
    by self write
    by * auth

access to attr=shadowLastChange
    by self write
    by * read

access to *
    by * read

# if no access controls are present, the default policy
# allows anyone and everyone to read anything but restricts
# updates to rootdn. (e.g., "access to * by * read")
#
# rootdn can always read and write EVERYTHING!

#####
# bdb database definitions
#####

loglevel 0
TLSCertificateFile /etc/ssl/servercerts/servercert.pem

```

```

TLSCACertificatePath /etc/ssl/certs/
TLSCertificateKeyFile /etc/ssl/servercerts/serverkey.pem
database bdb
suffix "dc=sanh,dc=uc,dc=edu"
rootdn "cn=Administrator,dc=sanh,dc=uc,dc=edu"
rootpw "{sha}CpobwMmwJ+MplURjvYYMctOkfINFRUJHUA=="
directory /var/lib/ldap
checkpoint 1024 5
cachesize 10000
index objectClass,uidNumber,gidNumber eq
index member,mail eq,pres
index cn,displayname,uid,sn,givenname sub,eq,pres

repllogfile /var/lib/ldap/repllog
replica host=10.78.3.244:389
        binddn="cn=Administrator,dc=sanh,dc=uc,dc=edu"
        bindmethod=simple
        credentials=password

```

A5. slapd.conf Secondary Server

```

# See slapd.conf(5) for details on configuration options.
# This file should NOT be world readable.
#
include /etc/openldap/schema/core.schema
include /etc/openldap/schema/cosine.schema
include /etc/openldap/schema/inetorgperson.schema
include /etc/openldap/schema/rfc2307bis.schema
include /etc/openldap/schema/yast.schema
include /etc/openldap/schema/samba3.schema
# Define global ACLs to disable default read access.

# Do not enable referrals until AFTER you have a working directory
# service AND an understanding of referrals.
#referral ldap://root.openldap.org

pidfile /var/run/slapd/slapd.pid
argsfile /var/run/slapd/slapd.args

# Load dynamic backend modules:
modulepath /usr/lib/openldap/modules
# moduleload back_ldap.la
# moduleload back_meta.la
# moduleload back_monitor.la
# moduleload back_perl.la

```

```

# Sample security restrictions
#   Require integrity protection (prevent hijacking)
#   Require 112-bit (3DES or better) encryption for updates
#   Require 63-bit encryption for simple bind
# security ssf=1 update_ssf=112 simple_bind=64

# Sample access control policy:
#   Root DSE: allow anyone to read it
#   Subschema (sub)entry DSE: allow anyone to read it
#   Other DSEs:
#       Allow self write access to user password
#       Allow anonymous users to authenticate
#       Allow read access to everything else
#   Directives needed to implement policy:
access to dn.base=""
    by * read

access to dn.base="cn=Subschema"
    by * read

access to attr=userPassword,userPKCS12
    by self write
    by * auth

access to attr=shadowLastChange
    by self write
    by * read

access to *
    by * read

# if no access controls are present, the default policy
# allows anyone and everyone to read anything but restricts
# updates to rootdn. (e.g., "access to * by * read")
#
# rootdn can always read and write EVERYTHING!

#####
# bdb database definitions
#####

loglevel 0
TLSCertificateFile /etc/ssl/servercerts/servercert.pem
TLSCACertificatePath /etc/ssl/certs/
TLSCertificateKeyFile /etc/ssl/servercerts/serverkey.pem
database bdb

```

```

suffix "dc=sanh,dc=uc,dc=edu"
rootdn "cn=Administrator,dc=sanh,dc=uc,dc=edu"
rootpw "{sha}CpobwMmwJ+MplURjvYYMctOkfINFRUJHUA=="
directory /var/lib/ldap
checkpoint 1024 5
cachesize 10000
index objectClass,uidNumber,gidNumber eq
index member,mail eq,pres
index cn,displayname,uid,sn,givenname sub,eq,pres

updatedn      "cn=Administrator,dc=sanh,dc=uc,dc=edu"
updateref     ldap://10.78.3.243

```

A6. smb.conf File

```

[global]
printing = cups
printcap name = cups
printcap cache time = 750
cups options = raw
printer admin = @ntadmin, root, administrator
security = user
encrypt passwords = yes
netbios name = Sandert1
passdb backend = ldapsam:"ldap://10.78.3.243 ldap://10.78.3.244"
guest ok = no
ldap admin dn = cn=Administrator,dc=sanh,dc=uc,dc=edu
ldap suffix = dc=sanh,dc=uc,dc=edu
ldap passwd sync = yes
ldap filter = (uid=%u)
[printers]
comment = All Printers
path = /var/tmp
printable = yes
create mask = 0600
browseable = no
guest ok = no
[print$]
comment = Printer Drivers
path = /var/lib/samba/drivers
write list = @ntadmin root
force group = ntadmin
create mask = 0664
directory mask = 0775
browseable = yes
guest ok = no

```

printable = no
[PublicShare]
path=/SecondShare/Public
read only = no
writable = yes
[homes]
comment = Home Directories
browseable = no
read only = No
inherit permissions = Yes
printable = no
[profiles]
comment = Network Profiles Service
path = %H
read only = No
store dos attributes = Yes
create mask = 0600
directory mask = 0700
browseable = no
printable = no

References

1. Carter, Gerald. "LDAP System Administration". California: O'Reilly, 2003.
2. Linux-HA Downloads. [Http://linux-ha.org/download/](http://linux-ha.org/download/). 2005.
3. Novel: SUSE Linux. [Http://www.novell.com/linux/suse/](http://www.novell.com/linux/suse/). 2005.
4. Openldap.org. [Http://www.openldap.org](http://www.openldap.org). 2005.
5. Samba.org. [Http://www.samba.org](http://www.samba.org). 2005.
6. Tolle, Troy. Sr Systems Engineer, College of Applied Science (Clifton Campus). Personal interview. October 6, 2004.
7. Ts, Eckstein, and David Collier-Brown. "Using Samba". California: O'Reilly, 2003.
8. Using OpenLDAP. [Http://www.metaconsultancy.com/whitepapers/ldap.htm](http://www.metaconsultancy.com/whitepapers/ldap.htm). 2005.