CIS225 Linux (UNIX) System and Network Administration

This is the second class dealing with UNIX type operating systems. The first was CIS137 Introduction to the UNIX Operating System which should have given you some of the history of UNIX type operating systems as well as learning how to edit files and use basic commands.

In this class, each student installs Debian Linux in a virtual environment as we proceed to learn more about the configuration files and various applications.

Debian GNU/Linux is a very rich environment with tens of thousands of applications available as well as a very robust package management system called aptitude. Many of you might now be using a Linux environment called Ubuntu which is a derivative of Debian.

Note: The Debian packages are named after characters in the movie “Toy Story.” There are usually 3 current versions, SID or “System In Development,” whatever is the current stable release, and the testing release which will be the next stable release.

At this time, Jessie is the testing release and Harry often introduces it to the class before it becomes the stable release.

We will explore some of the most common uses of Linux in a server environment before eventually moving to the desktop graphical environment.

This allows you to learn where various files are located and how to edit them to make things work properly. Many of these operations are hidden in the graphical environment so it is best that you learn to work from the command line before moving to a graphical environment.

Some of the common applications that we will deal with include Apache, the web server, Bind9, the DNS server, Squid, the proxy server, Samba, the file sharing server, as well as various other topics such as file systems and interacting with other network devices.

Typically we also spend several class sessions dealing with compiling a kernel to optimize it for our particular hardware. This is not something you might commonly do but it is invaluable experience in case you need to do it in the future.

These are notes that I have assembled over various semesters of the CIS225 Linux class.

Not every topic is always covered in every class and often the instructors will cover the topics in more detail with much more background information. They are meant to just be a guideline and where IP addresses are noted, you must use the ones given in class.

I do occasionally update them when I find errors or to add more information. Please note that as of summer 2013, the classrooms have been configured to use static to assign IP addresses. We will manually configure them.

Sometimes the CIS225 class has two instructors, Harry McGregor and myself, George Cohn. This is done to allow both of us to teach more credit hours and provides flexibility if one or the other instructor is unavailable.

We have also shared other classes such as the CIS-218 - Introduction to Voice over IP class which is also based on Linux. I also teach Windows and Server 2012 classes while Harry teaches the advanced Linux class.

The various notes for exercises are in no particular order and some topics may be covered in more or less detail, depending on class interests.

Obviously we have to install Linux first. We are using a virtualization program called VM Workstation. Various versions of VM Ware are available, often as a free download if you wish to set something similar up on a computer at home. Also, VirtualBox from Oracle is available as a free download but does not have as great of video support as VM Ware but is totally useful.

Typically in class we will already have a template created in VM Workstation and when you first start the virtual machine, it will boot the install image. The install image is a small iso file called Debian Business Card and is available as a free download from http://www.debian.org.

We will walk you through the installation but it generally consists of answering on screen questions and creating passwords for the root user and the standard user.

There are a couple of very useful tools that we probably want to install right away. One is the VIM or VI iMproved editor. The version of VI that comes with Debian by default has some confusing commands. The other tool is GPM or general purpose mouse which allows us to highlight and paste much like in Windows.

To install these two programs, run the following command:
apt-get install vim gpm This will go out to our repository mirror, download the appropriate applications and any required dependencies and automatically install them.

DNS Server

Once Debian is installed, we generally move on to installing the DNS server application, Bind9. DNS is the service which converts human readable names like google.com into the proper IP addresses that all computers need to communicate with each other.

First we need to install Bind9:
apt-get install bind9

Once Bind9 is installed, we need to edit two files to make it work properly. Run the following command to edit the first file: **vi /etc/bind/named.conf.options**

This will open the named.conf.options file in the vi editor. Go down to about line 13 where you will see something like this:

```
// forwarders {
// 0.0.0.0;
// };
```

The // characters are used to mark a line as a comment. Hit the **I** key to get into insert mode and remove them from in front of the three lines shown above. Now change the 0.0.0.0 to 10.227.4.5 This is the ip address of our DNS server in this classroom.

Because we are now using the Wheezy version of Debian and it supports ipv6, you will have to comment out this line by adding the // in front of it:

```
// dnssec-validation auto;
```

Now hit the **ESC** key, the **:** key and enter **wq** From now on, I will show this as **esc:wq**

What this does is get the attention of the vi or vim editor and write and quit the file. Toward the end of these notes is a cheat sheet showing many more of the vi commands.

What you have just done is configured the Bind9 DNS program to forward all requests for name to ip resolution that it doesn't know about to the next available DNS server which is located at 10.227.4.5.

Now we need to set up our own machine so it will use itself as a forwarder. To do this, we need to edit the /etc/resolv.conf file. Run the following command: vi /etc/resolv.conf

This will open the resolv.conf file. You will see that it already has two entries, the top should be something like a227.cis and the bottom should be our nameserver 10.227.4.5

What we need to do here is add our internal loopback address in between these two entries. Note: all computers generally have the same loopback address: 127.0.0.1 This is the internal IP for your machine.

So edit the file so it looks something like this:

```
search cis225.a227.cis
nameserver 127.0.0.1
nameserver 10.227.4.5
```

esc:wq to save.

Anytime we make a change to a configuration file in Linux, we usually need to restart the application to reload the new configuration file. In this case, this takes the form of
/etc/init.d/bind9 restart

What this command is telling us is to run the /etc/init.d/name_of_application restart to restart the program and read the new configuration file. Contrast this to Windows which often makes you restart the entire computer when making a change to a program.

At this point you should have a working DNS server and your machine should be using itself to do lookups. You can test it by running hostname a227.cis and it should return with the proper IP address.

When you installed Debian, once your network settings were correct, it automatically picked up your host name. This is because Harry has set up the DNS server on R2D2 to do a reverse look up of IP address to hostname.

A sample configuration of how this is done is shown in the separate bind9.pdf file.

Apache web server

Without a doubt, Apache is the most often used web server on the Internet. Estimates run as high a 85% of all web servers are running Apache. There are a few others like nginx for Linux and of course Windows has its IIS (Internet Information Server.) However, there is even a version of Apache for Windows. Visit the Apache website at http://www.apache.org to see more details about the Apache foundation and its projects.

apt-get install apache2

Once we install apache, we need to edit the default web page a bit.

The default web page is located at /var/www and is called index.html

Note: *In Jessie, they created another subdirectory below www called html and that is where the web pages are located. Also, they broke up the configuration files into smaller pieces.*

We can edit it to add our name, our machine number, and our IP address. This makes it easy to identify which machine we are viewing.

If you are providing virtual hosting for a number of web sites, we will want to install the ftp server to allow our users to upload their web content without having physical access to the server. Most ISP’s that provide paid web hosting do something like this although they may use a “Control Panel” to allow you to upload instead of having to install an FTP client on you Windows or Linux desktop.

VSFTP Secure FTP server.

Apt-get install vsftpd

Also note that we sometimes need an ftp server to hold configuration files for other devices. For example, the Polycom VoIP phones that we use in the Cis 218 class can be totally configured by downloading various xml files from an ftp or tftp server. TFTP or trivial file transfer protocol is not secure and does not require a username or password. The more secure method is to use ftp and create a folder for the configuration files at /home/PlcmSlIp.

Also note that we can set up vsftpd to chroot or automatically restrict users to their own folder and not let them traverse other folders where they could do accidental or malicious damage.

Below is a sample config file for vsftpd. We have to make a few changes to allow our users to log in and write to their folders. Compare this with the default config file as created when you installed vsftpd.

```
#Example config file /etc/vsftpd.conf
#
# The default compiled in settings are fairly paranoid. This sample file
# loosens things up a bit, to make the ftp daemon more usable.
# Please see vsftpd.conf.5 for all compiled in defaults.
#
# READ THIS: This example file is NOT an exhaustive list of vsftpd options.
# Please read the vsftpd.conf.5 manual page to get a full idea of vsftpd's
# capabilities.
#
#
# Run standalone?  vsftpd can run either from an inetd or as a standalone
# daemon started from an initscript.
listen=YES
#
# Run standalone with IPv6?
# Like the listen parameter, except vsftpd will listen on an IPv6 socket
# instead of an IPv4 one. This parameter and the listen parameter are mutually
# exclusive.
#listen_ipv6=YES
#
# Allow anonymous FTP? (Beware - allowed by default if you comment this out).
anonymous_enable=NO
#
# Uncomment this to allow local users to log in.
local_enable=YES
#
# Uncomment this to enable any form of FTP write command.
write_enable=YES
#
# Default umask for local users is 077. You may wish to change this to 022,
# if your users expect that (022 is used by most other ftpd's)
local_umask=022
#
# Uncomment this to allow the anonymous FTP user to upload files. This only
# has an effect if the above global write enable is activated. Also, you will
# obviously need to create a directory writable by the FTP user.
#anon_upload_enable=YES
#
```

```
# Uncomment this if you want the anonymous FTP user to be able to create
# new directories.
#anon_mkdir_write_enable=YES
#
# Activate directory messages - messages given to remote users when they
# go into a certain directory.
dirmessage_enable=YES
#
# If enabled, vsftpd will display directory listings with the time
# in  your  local  time  zone.  The default is to display GMT. The
# times returned by the MDTM FTP command are also affected by this
# option.
use_localtime=YES
#
# Activate logging of uploads/downloads.
xferlog_enable=YES
#
# Make sure PORT transfer connections originate from port 20 (ftp-data).
connect_from_port_20=YES
#
# If you want, you can arrange for uploaded anonymous files to be owned by
# a different user. Note! Using "root" for uploaded files is not
# recommended!
#chown_uploads=YES
#chown_username=whoever
#
# You may override where the log file goes if you like. The default is shown
# below.
#xferlog_file=/var/log/vsftpd.log
#
# If you want, you can have your log file in standard ftpd xferlog format.
# Note that the default log file location is /var/log/xferlog in this case.
#xferlog_std_format=YES
#
# You may change the default value for timing out an idle session.
idle_session_timeout=600
#
# You may change the default value for timing out a data connection.
data_connection_timeout=1200
#
# It is recommended that you define on your system a unique user which the
# ftp server can use as a totally isolated and unprivileged user.
#nopriv_user=ftpsecure
#
# Enable this and the server will recognise asynchronous ABOR requests. Not
# recommended for security (the code is non-trivial). Not enabling it,
# however, may confuse older FTP clients.
#async_abor_enable=YES
#
# By default the server will pretend to allow ASCII mode but in fact ignore
# the request. Turn on the below options to have the server actually do ASCII
# mangling on files when in ASCII mode.
# Beware that on some FTP servers, ASCII support allows a denial of service
# attack (DoS) via the command "SIZE /big/file" in ASCII mode. vsftpd
# predicted this attack and has always been safe, reporting the size of the
# raw file.
# ASCII mangling is a horrible feature of the protocol.
#ascii_upload_enable=YES
#ascii_download_enable=YES
#
# You may fully customise the login banner string:
ftpd_banner=All Logons are moniterd.  Unauthorized users will be prosecuted.
#
# You may specify a file of disallowed anonymous e-mail addresses. Apparently
# useful for combatting certain DoS attacks.
#deny_email_enable=YES
# (default follows)
#banned_email_file=/etc/vsftpd.banned_emails
#
# You may restrict local users to their home directories.  See the FAQ for
# the possible risks in this before using chroot_local_user or
# chroot_list_enable below.
#chroot_local_user=YES
#
# You may specify an explicit list of local users to chroot() to their home
# directory. If chroot_local_user is YES, then this list becomes a list of
# users to NOT chroot().
#chroot_local_user=YES
#chroot_list_enable=YES  # If this is uncommented, users can only access their own folder.
# (default follows)
#chroot_list_file=/etc/vsftpd.chroot_list  # You need to put their username in this list.
#
# You may activate the "-R" option to the builtin ls. This is disabled by
# default to avoid remote users being able to cause excessive I/O on large
# sites. However, some broken FTP clients such as "ncftp" and "mirror" assume
# the presence of the "-R" option, so there is a strong case for enabling it.
#ls_recurse_enable=YES
#
# Debian customization
#
# Some of vsftpd's settings don't fit the Debian filesystem layout by
# default.  These settings are more Debian-friendly.
```

```
#
# This option should be the name of a directory which is empty.  Also, the
# directory should not be writable by the ftp user. This directory is used
# as a secure chroot() jail at times vsftpd does not require filesystem
# access.
secure_chroot_dir=/var/run/vsftpd/empty
#
# This string is the name of the PAM service vsftpd will use.
pam_service_name=vsftpd
#
# This option specifies the location of the RSA certificate to use for SSL
# encrypted connections.
rsa_cert_file=/etc/ssl/private/vsftpd.pem
```

Below are some commonly used commands used with Debian and most other linuxes.

Install & update commands – specific to Debian aptitude package manager
apt-get update – updates list for programs installed on your computer
apt-get upgrade – does the actual upgrade using the database from above command
apt-get clean – run after the above commands to remove update file – saves disk space
apt-get install (package name) installs application IE: apt-get install vim installs vim text editor

shutdown –h now shuts down your Linux operating system.

If you specify a number instead of **now**, it waits that many minutes before shutting down.

Why would we want to specify a time rather than just now? Windows clients like Windows 7 can have many user accounts but only one user can be logged in at a time and using the system.

Linux is a true multi-user operating system and many users can be logged into one server using different applications. By specifying a time, we send out a message to all logged on users alerting them to the fact that the server will be going down in X minutes. This gives them a chance to save their work and log off before the server shuts down.

User management commands – applies to most distributions of Linux

adduser (username) adds a user to your system and creates their home directory in the /home folder
passwd by itself prompts you for a new password for the root
passwd (**username**) prompts you for a new password for the user. In both cases above, don’t enclose in parenthesis.
deluser would delete a user from the system but leave all of their mail and files intact.
deluser --remove-all-files would delete them completely from the system. We will discuss why you would use one option or the other in class.

Browsers
w3m (URL) is a text based Internet browser. It works but is not very intuitive for browsing
Lynx is a another text based browser that is functional but again not the greatest.

Either command and localhost or your ip address will display the /var/www/index.html page if you have the Apache2 web server installed. (/var/www/html/index.html on Jessie)

IceWeasel is the Debian version of Firefox
IceDove is the Debian version of Thunderbird the mail program.

Working with files

You can edit files without going to their subdirectory if you specify the full path.
IE: vi \var\www\index.html will open the index.html file in the \var\www folder for editing.

vi commands that are handy are I for insert which allows you to start editing, esc : (colon) wq tells it to write the changes and quit. To exit without saving, esc : q! You can turn syntax highlighting on to make editing easier with esc : syn on

The vi editor that comes by default with Debian is a little strange. A better version is vim (vi improved). You can install it with **apt-get install vim**.

nano is a good wysiwyg editor that is reasonably friendly. You use Alt O to save a file after changes and Alt X to exit. If you have made changes and hit Alt X, it will prompt you if you need to save before exiting. For nano, you turn syntax highlighting on and off with Alt y

Directory commands

cd /directory changes to specified directory. To change to another directory under that one, use cd directory without the / IE: you are in etc directory and want to get to networks directory under it, just enter cd networks and hit return.

ls is the directory listing command. ls –l gives you more detail about the files such as their permissions. ls –al will do the same thing but show hidden directories as well. Hidden directories start with the . (dot) character.

cp is the copy file command, IE: cp (Filename) /home/user/ would copy a file from current directory to /home/users directory.

mv is the rename command. mv (filename1) (filename2) would rename the first file.

rm (filename) deletes that file. rm –r will recursively delete all files in that directory, use carefully!

File permissions

All files and directories in Linux have permissions. The permissions are read, write and execute. If you look at a file using ls –l and it shows rwxr—r—that means that the owner has the ability to read write and execute the file, the r—in the middle means anyone in that group can read but not write or execute the file, and the third r—means that anyone else (other) can read but not write or execute the file.

Every file on your Linux system, including directories, is owned by a specific user and group. Therefore, file permissions are defined separately for users, groups, and others.

User: The username of the person who owns the file. By default, the user who creates the file will become its owner.

Group: The usergroup that owns the file. All users who belong to the group that owns the file will have the same access permissions to the file.

Other: A user who isn't the owner of the file and doesn't belong in the same group the file does. In other words, if you set a permission for the "other" category, it will affect everyone else by default. For this reason, people often talk about setting the "world" permission bit when they mean setting the permissions for "other."

There are several commands that allow you to change the permissions of a file.

IE: chmod (username) filename would change a file to be owned by that user. chgrp (username) filename would change a file so that anyone in that group that had the appropriate permissions could work with the file

.

File permissions are based on a binary number system. If a file has 777 permissions, that means anyone can read, write or execute the file. 644 would mean the filename owner could read and write, and the group would be able to read and the other group could read.

To better illustrate this: The binary system looks like this:
4 2 1 so if bits 4 and 2 are ones, then = 6

The example of 644 would look like this:

Binary = 110100100 own grp any

6 4 4

rw- r-- r—filename

Exercise in creating a password protected directory in Apache2

Advanced Apache configurations

In this exercise you will edit the default configuration file and create an htaccess file for storing the user name and password.

The htaccess file can be called anything and stored anywhere on the hard drive. You just need to make sure your path points to it in the default configuration file.

The htaccess file takes the form of username:encrypted_password For example, class:UuKiRO880IkWg is for a username of class and a password of 123456

Create this by opening your favorite editor in the directory where you want to save the file and type class:UuKiRO880IkWg and save as htaccess. If you save it as .htaccess, it will be a hidden file.

Make a directory under /var/www called protect mkdir /var/www/protect

Copy the index file from /var/www to /var/www/protect

cp /var/www/index.html /var/www/protect/

Edit the /var/www/protect/index.html file to add something like: This is a password protected folder.

Now edit the /etc/apache2/sites-available/default file as follows, anything preceeded by a # is a comment and ignored by the server.

```
# *********** Virtual Host ****************
<VirtualHost *:80> # Use IP address of your machine
ServerAdmin user@mail.com # Email address of web server administrator
ServerName your_server.com # This would be name of your web site
DocumentRoot /var/www
ServerSignature Off # This turns off information that identifies OS

<Directory />
Options FollowSymLinks
AllowOverride None
</Directory>

<Directory /var/www/>
Options Indexes FollowSymLinks MultiViews
AllowOverride None
Order allow,deny
allow from all
```

```
</Directory>
ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/

<Directory "/usr/lib/cgi-bin">
AllowOverride None
Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
Order allow,deny
Allow from all
</Directory>

ErrorLog ${APACHE_LOG_DIR}/error.log
# Possible values include: debug, info, notice, warn, error, crit,
# alert, emerg.
LogLevel warn
CustomLog ${APACHE_LOG_DIR}/access.log combined

Alias /docs /usr/share/doc/apache2-doc/manual # Docs are not installed by default

<Directory /usr/share/doc/apache2-doc/manual>
Options +Indexes FollowSymLinks MultiViews
AllowOverride AuthConfig Options FileInfo Limit
Order allow,deny
Allow from all
</Directory>

# The following is added to allow access to password protected directory

Alias /protect /var/www/protect # Points to folder that is password protected

<Directory /var/www/protect>
AuthType Basic
AuthName "Authorization Required"
AuthUserFile /home/user/htaccess # Points to folder where password file exists
Require valid-user # change user to name of your home folder
</Directory>

</VirtualHost>


# ********** virtual host **********
```

Save default file and run /etc/init.d/apache2 restart to reload configuration file. If you have a typo in the file, it will fail but give you a hint as to the line the error is on.

If it runs correctly, the apache web server is started and you can surf to either localhost/protect or IP_address/protect and it will prompt you to enter user name and password to access page.

Apache2 with SSL Certificates

Sometimes we need a secure encrypted web page, especially if we are dealing with sensitive information or credit card numbers. This is done using SSL. In commercial web browsers like IE and Firefox, there are embedded “Root Certificates” issued by companies like Thawte, Veri-Sign, Microsoft, etc. If you have a certificate from one of these recognized providers installed on your server, by third party trust, they are trusted because of the embedded root certificates.

This allows a user to visit an https web site on your server and not be warned about an invalid certificate, as long as your certificate is correct, has not expired, or been revoked.

These certificates come at a cost and depending on the level of verification, range from nearly free to thousands of dollars per year. For that reason, we will create what is called a “Self-Signed Certificate” for this class as it demonstrates the principles, but you would need to purchase a certificate if you intended to do e-commerce, etc.

First we need to enable the SSL module for apache2

A2enmod ssl You will get some messages including one on how to create a self-signed certificate.

Apache2 comes with a default-ssl configuration file which we will modify.

We need to create a folder for the home directory for our secure web site and add some content to it.

mkdir /var/www/secure creates a folder under /var/www/ called secure.
cp /var/www/index.html /var/www/secure/ will copy the default index.html file in the non-secure web page to the new home directory. Now we need to edit it a bit to distinguish it from the regular home page.

vi /var/www/secure/index.html Add a line somewhere that says This is the Secure Web Site
Save and exit and restart apache /etc/init.d/apache2 restart

Now if I surf to https://IP_Address of a machine, it should prompt me for a security warning about the existing certificate not being valid. I will demonstrate and explain this is class.

Debian by default comes with a certificate called “snakeoil” for testing purposes. We will now create a self signed certificate to replace it. We will still get a warning but it demonstrates how certificates are generated and used.

First we need to make sure openssl is installed so apt-get install openssl If it is not installed, go ahead and install it.

We Begin by generating a private key:

```
openssl genrsa -out mycert.key 1024
```

Next, generate a certificate request and enter the information:

```
openssl req -new -key mycert.key -out mycert.csr
```

It will ask you to anser some questions:

```
Country Name (2 letter code) [US]:
State or Province Name (full name) [AZ]:
Locality Name (eg, city) [Tucson]:
Organization Name (eg, company) [Usually the web site]:
Organizational Unit Name (eg, section) [CIS 225 Class]:
Common Name (eg, YOUR name) [George Cohn]:
Email Address [gwcohn@pima.edu]:
```

Skip the optional information

Next, generate the self-signed certificate. You can specify the number of days the cert is valid.

The default is 365 days or one year.

```
openssl x509 -req -days 365 -in mycert.csr -signkey mycert.key -out mycert.cert
```

You no longer need the .csr request file. It would be sent to a CA to request a paid certificate.

Create a folder and move the .key and .cert files into it:

```
mkdir /etc/apache2/ssl
mv *.cert /etc/apache2/ssl
mv *.key /etc/apache2/ssl
chmod 400 /etc/apache2/ssl/*.key This makes the file rw only by root.
```

You'll need to configure the SSL settings for the site:

```
vi /etc/apache2/sites-eavailable/default-ssl
SSLCertificateFile /etc/apache2/ssl/mycert.cert
SSLCertificateKeyFile /etc/apache2/ssl/mycert.key
```

Save and exit and restart apache

```
/etc/init.d/apache2 restart
```

We now need to enable the ssl web site. Run the command a2ensites default-ssl This enables the secure web site. Now run /etc/init.d/apache2 restart to reload the configuration for Apache

Now when we browse to our secure web site, we will still get a warning but it shows our certificate now. The process is similar for applying for a trusted certificate from a recognized certificate authority.

Setting up Exim4, Dovecot, and Squirrelmail on Debian Squeeze

Since Exim4 is already installed, we need to make a configuration change for it to use the "Maildir" format.

Run this command from the root prompt: **dpkg-reconfigure exim4-config**

I will walk you through the choices but we want it to deliver mail to the users "Maildir."

Restart Exim4 **/etc/init.d/exim4 restart** This reloads the configuration file.

If you vi /etc/exim4/update-exim4.conf.conf you should see this at the bottom of the configuration file: dc_localdelivery='maildir_home'

Exit from vi with Esc:q!

Next we want to install the Dovecot IMAP e-mail program.

apt-get install dovecot-imapd

vi /etc/dovecot/dovecot.conf

We need to make a couple of changes here.

If you **Esc : 25** it will take you to line 25 where we need to uncomment the protocols = imap imaps line

This allows Dovecot to serve as an imap mail client.

Next **Esc : 47** to go to line 47. We need to uncomment the line listen = *

This allows it to listen on any port.

Now **Esc : 224** to go to line 224. We need to uncomment the line mail_location = maildir:~/Maildir

This allows it to use the Maildir format.

Now **Esc : wq** to save the file and restart Dovecot with **/etc/init.d/dovecot restart**

This reloads the configuration for dovecot.

Now we need to install squirrelmail which is a web based mail client.

apt-get install squirrelmail

From the root command line run **squirrelmail-configure**

This will present an interactive menu where SquirrelMail can be configured.

Select option, D. Set pre-defined settings for specific IMAP servers. This preloads some settings specifically for your IMAP server package.

Enter dovecot

Press S to save and Q to quit.

Under /etc/squirrelmail/ there is a file called apache.conf In order to be able to access squirrelmail from a web browser, we need to let apache2 know about this file.
vi /etc/apache2/apache2.conf and add this line at the bottom of the file:

```
include /etc/squirrelmail/apache.conf
```

This tells apache2 to include this configuration file so we can access squirrelmail at http://(IP address)/squirrelmail

We need to restart apache2 to reload the configuration
/etc/init.d/apache2 restart

Now open the iceweasal browser and type in http://your_ip_address/squirrelmail

It should display the web interface and you can log in as your standard user (not root) and your user password. You will see a full featured web page that allows you to read and send mail for your normal user.

Send Email via Telnet

On occasion, we write code that sends email. Sometimes, it actually works the first time. More often, we need to figure out why not.

While it's nice to have the computer emulate what the human would get bored doing, sometimes it helps to have the human do what the computer can't quite seem to do on its own.

For that reason, we sometimes resort to manual telnet sessions with a remote mail server.

Simple manual telnet session with mail host Uppercase/lowercase does not appear to be significant.

You type this

Telnet to hostname on port 25

220 (then identifies itself - possibly with several lines of 220 + text)

HELO your_domain_name or whatever

250 (followed by human readable message)

MAIL FROM:you@hostname.com (ie, your email address)

250 is syntactically correct (or similar)

RCPT TO:them@someplace_else.com (email address you want to send to)

250 is syntactically correct

DATA

Tells you to send data then CRLF period CRLF at end

You type your message then CRLF period CRLF (ie, type a period on a line by itself then hit ENTER)

250

QUIT

Signoff message

Squid Proxy Server

The squid proxy server application is typically installed to cache web pages although it has many other uses as well. Why would we want to cache web pages? It speeds up your Internet browsing as it caches pages in memory and when you go to the same page, it does not have to go to the Internet to get it.

For a small business sharing a dsl connection, this makes the speed of browsing the Internet reasonably fast.

A web developer may not want you to cache a web page it if it contains rapidly changing content. They can embed a piece of code in their page to prevent caching.

```
<html>
<head>
<meta http-equiv="Pragma" content="no-cache">
<!-- Pragma content set to no-cache tells the browser not to cache the page
This may or may not work in IE -->
```

For the purpose of this class, we will install a simple proxy server and demonstrate how it can be used to block certain content.

apt-get install squid

This will install squid and a couple of dependencies. We then need to edit the squid.conf file to our needs.

vi /etc/squid/squid.conf

Esc : 604 will take us to about where we need to make the initial edit. Hit I to enable editing.

Note: line numbers are approximate and depend on if the file has been edited before. Use the numbers given to locate the general area.

Under the line acl to_localhost dst 127.0.0.0/8 0.0.0.0/32 add this line:

Acl pimanet src 10.227.15.0/24

This tells our server to use a network we are calling pimanet at IP address 10.227.15.0 with a subnet mask of 255.255.255.0 as our outgoing connection. The /24 is CIDR notation for 255.255.255.0

Now **Esc : 673** and it will take you about to the next line to edit. Hit I to enable editing.

Under the line #http_access deny to_localhost add this line:

http_access allow pimanet

This allows our server to use the access control entry we added above. Now restart the proxy server with /etc/init.d/squid restart to load the new configuration. The server is now active.

You must tell your browser to use the proxy server to connect to the Internet. In Firefox this is under Tools/options/advanced/network tab/settings Select manual proxy and enters the IP address of your server and port 3128 in the port box. Click OK a couple of times to save your Firefox settings.

Now when you surf to the Internet, it will connect to your server first then the web page you are seeking. It will store a copy of that page in proxy server memory for the next time that you go to it.

Note: Due to the way networking is configured in the classroom, this does not work on your machine so I will use a laptop with Windows to demonstrate.

This is a perfectly usable server but let's see how we can also use it to control what web sites our users can visit. Let's assume you are a company and you have noticed that many employees are surfing facebook.com on company time. Let's block this web site.

vi /etc/squid/squid.conf and **Esc : 606** This should take us about to the line where we added the

acl pimanet src 10.227.15.0/24 Hit I to edit and under that line enter this:

```
acl blocksites dstdomain .facebook.com
```

What this line is telling us is to use an access control list called blocksites to block facebook.com. The dot before facebook.com tells it to block any subdomains of facebook.com as well.

Now add this line below that one:

http_access deny blocksites That line tells our proxy server to use the acl we created above.

Esc :wq to save and restart squid /etc/init.d/squid restart Now surf to www.facebook.com and you will get a decidedly unfriendly message telling you it is denied. To forestall arguments, we can change the displayed message as well to explain why it is blocked.

The error messages for squid are in the folder /usr/share/squid/errors/English/ We will now create a custom error message.

Enter nano /usr/share/squid/errors/English/ERR_BLOCKED_SITES This will open a blank document where we can add the following:

```
<HTML>
<HEAD>
<TITLE>ERROR: Blocked content</TITLE>
</HEAD>
<BODY>
<center>
<H1>Web Site is blocked due to company IT policy</H1>
<p>Please contact the helpdesk for more information :</p>
Phone: 349-0859<br>
Email: gwcohn@pima.edu<br>
```

Use any phone number and e-mail address you desire. Hit Ctrl O to save and Ctrl X to exit.

Now we need to go back to the squid.conf file and tell it to use this error message.

vi /etc/squid/squid.conf Esc : 610 and add this line right under the

acl blocksites dstdomain .facebook.com line

```
deny_info ERR_BLOCKED_SITES blocksites
```

Save with **Esc :wq**

Run **/etc/init.d/squid restart** to reload the configuration

This gives our users an informative message indicating that we have blocked this site for a company policy reason.

Squid can also be used to block specific files such as mp3, jpg, WMV, etc.
In a corporate environment this may be used to prevent users from downloading tunes, movies, etc that may violate copyright laws and place the company at legal risk.

The process is similar to what we did above and you can visit the squid web site at http://www.squid-cache.org for many more examples of how to use squid not only for controlling outgoing access but also limiting incoming access to protect your client machines from malware.

CUPs and Linux Printing

CUPS (Common Unix Printing System) provides an easy way to manage printers in the Linux environment.

You can administer them from a GUI or a command line.

```
apt-get install cups cups-client cupd-pdf
```

If you do not have a GUI, you can use a text based browser.

```
apt-get install lynx-cur
```

We need to make a few changes to our cupsd.conf file if we want to be able to access it from another computer on the network.

See cupsd.conf printout.

Now we can browse to localhost:631 by invoking lynx localhost:631

This opens a management window to CUPS which will allow us to install a printer.

Use the up-down arrow keys to navigate. The options will be highlighted when you stop on one.
CUPS needs to set a cookie so answer yes when prompted

When it prompts you for a user name and password, use root and your password.

We will be installing two print options, the laser printer and a pdf file printer. We will do the laser jet first then add the PDF printer later.

Go through print setup. HP printer probably uses format http://IP_address /ipp

Now that we have a printer installed, we can look at some of the command line options for printing. Explore command line options.

Now install PDF printing apt-get install cups-pdf

Now we can go back and add a virtual pdf printer. Is we set it so the regular user can use it, it will create PDF files in the users home/PDF/ directory. For root, it creates the PDF files in the PDF folder in roots home

```
# CUPS Configuration (cupsd.conf )
# Sample configuration file for the CUPS scheduler. See "man cupsd.conf" for a
# complete description of this file. Note IP address changes in 4 locations
#
# Log general information in error_log - change "warn" to "debug"
# for troubleshooting...
LogLevel warn
# Deactivate CUPS' internal logrotating, as we provide a better one, especially
# LogLevel debug2 gets usable now
MaxLogSize 0
# Administrator user group...
SystemGroup lpadmin
# Only listen for connections from the local machine.
Listen localhost:631
Listen 192.168.0.6:631 # Add this line. Change to your machine IP address
# Allows access from your machine IP address
Listen /var/run/cups/cups.sock
# Show shared printers on the local network.
Browsing On
BrowseOrder allow,deny
BrowseAllow all
BrowseLocalProtocols CUPS dnssd
# Default authentication type, when authentication is required...
DefaultAuthType Basic
# Restrict access to the server...
<Location />
Order allow,deny
Allow localhost
Allow 192.168.0.* # Allows access from local network .
#Use first 3 octets of your IP address
</Location>
# Restrict access to the admin pages...
<Location /admin>
Order allow,deny
Allow localhost
Allow 192.168.0.* # Allows access from local network.
#Use first 3 octets of your IP address
</Location>
# Restrict access to configuration files...
<Location /admin/conf>
AuthType Default
Require user @SYSTEM
Order allow,deny
Allow localhost
Allow 192.168.0.* # Allows access from local network.
# Use first 3 octets of your IP address
</Location>
# Set the default printer/job policies...
<Policy default>
# Job-related operations must be done by the owner or an administrator...
<Limit Send-Document Send-URI Hold-Job Release-Job Restart-Job Purge-Jobs Set-Job-
Attributes Create-Job-Subscription Renew-Subscription Cancel-Subscription Get-Notifications
Reprocess-Job Cancel-Current-Job Suspend-Current-Job Resume-Job CUPS-Move-Job CUPS-
Get-Document>
Require user @OWNER @SYSTEM
Order deny,allow
</Limit>
# All administration operations require an administrator to authenticate...
<Limit CUPS-Add-Modify-Printer CUPS-Delete-Printer CUPS-Add-Modify-Class CUPS-
Delete-Class CUPS-Set-Default CUPS-Get-Devices>
AuthType Default
Require user @SYSTEM
Order deny,allow
</Limit>
# All printer operations require a printer operator to authenticate...
<Limit Pause-Printer Resume-Printer Enable-Printer Disable-Printer Pause-Printer-After-
Current-Job Hold-New-Jobs Release-Held-New-Jobs Deactivate-Printer Activate-Printer
Restart-Printer Shutdown-Printer Startup-Printer Promote-Job Schedule-Job-After CUPS-
Accept-Jobs CUPS-Reject-Jobs>
AuthType Default
Require user @SYSTEM
Order deny,allow
</Limit>
# Only the owner or an administrator can cancel or authenticate a job...
<Limit Cancel-Job CUPS-Authenticate-Job>
Require user @OWNER @SYSTEM
Order deny,allow
</Limit>
<Limit All>
Order deny,allow
</Limit>
```

```
</Policy>
# Set the authenticated printer/job policies...
<Policy authenticated>
# Job-related operations must be done by the owner or an administrator...
<Limit Create-Job Print-Job Print-URI>
AuthType Default
Order deny,allow
</Limit>
<Limit Send-Document Send-URI Hold-Job Release-Job Restart-Job Purge-Jobs Set-Job-
Attributes Create-Job-Subscription Renew-Subscription Cancel-Subscription Get-Notifications
Reprocess-Job Cancel-Current-Job Suspend-Current-Job Resume-Job CUPS-Move-Job CUPS-
Get-Document>
<Limit Send-Document Send-URI Hold-Job Release-Job Restart-Job Purge-Jobs Set-Job-
Attributes Create-Job-Subscription Renew-Subscription Cancel-Subscription Get-Notifications
Reprocess-Job Cancel-Current-Job Suspend-Current-Job Resume-Job CUPS-Move-Job CUPS-
Get-Document>
AuthType Default
Require user @OWNER @SYSTEM
Order deny,allow
</Limit>
# All administration operations require an administrator to authenticate...
<Limit CUPS-Add-Modify-Printer CUPS-Delete-Printer CUPS-Add-Modify-Class CUPS-
Delete-Class CUPS-Set-Default>
AuthType Default
Require user @SYSTEM
Order deny,allow
</Limit>
# All printer operations require a printer operator to authenticate...
<Limit Pause-Printer Resume-Printer Enable-Printer Disable-Printer Pause-Printer-After-
Current-Job Hold-New-Jobs Release-Held-New-Jobs Deactivate-Printer Activate-Printer
Restart-Printer Shutdown-Printer Startup-Printer Promote-Job Schedule-Job-After CUPS-
Accept-Jobs CUPS-Reject-Jobs>
AuthType Default
Require user @SYSTEM
Order deny,allow
</Limit>
# Only the owner or an administrator can cancel or authenticate a job...
<Limit Cancel-Job CUPS-Authenticate-Job>
AuthType Default
Require user @OWNER @SYSTEM
Order deny,allow
</Limit>
<Limit All>
Order deny,allow
</Limit>
</Policy>
```

#

Vi or Vim (V (i)mproved) Cheat Sheet

Cursor movement

h - move left
j - move down
k - move up
l - move right
w - jump by start of words (punctuation considered words)
W - jump by words (spaces separate words)
e - jump to end of words (punctuation considered words)
E - jump to end of words (no punctuation)
b - jump backward by words (punctuation considered words)
B - jump backward by words (no punctuation)
0 - (zero) start of line
^ - first non-blank character of line
$ - end of line
G - Go To command (prefix with number - 5G goes to line 5)

Note: Prefix a cursor movement command with a number to repeat it. For example, 4j moves down 4 lines.

Insert Mode - Inserting/Appending text

i - start insert mode at cursor
I - insert at the beginning of the line
a - append after the cursor
A - append at the end of the line
o - open (append) blank line below current line (no need to press return)
O - open blank line above current line
ea - append at end of word
Esc - exit insert mode

Editing

r - replace a single character (does not use insert mode)
J - join line below to the current one
cc - change (replace) an entire line
cw - change (replace) to the end of word
c$ - change (replace) to the end of line
s - delete character at cursor and subsitute text
S - delete line at cursor and substitute text (same as cc)
xp - transpose two letters (delete and paste, technically)
u - undo
. - repeat last command

Marking text (visual mode)
 v - start visual mode, mark lines, then do command (such as y-yank)
 V - start Linewise visual mode
 o - move to other end of marked area
 Ctrl+v - start visual block mode
 O - move to Other corner of block
 aw - mark a word
 ab - a () block (with braces)
 aB - a {} block (with brackets)
 ib - inner () block
 iB - inner {} block
 Esc - exit visual mode

Visual commands
 > - shift right
 < - shift left
 y - yank (copy) marked text
 d - delete marked text
 ~ - switch case

Cut and Paste
 yy - yank (copy) a line
 2yy - yank 2 lines
 yw - yank word
 y$ - yank to end of line
 p - put (paste) the clipboard after cursor
 P - put (paste) before cursor
 dd - delete (cut) a line
 dw - delete (cut) the current word
 x - delete (cut) current character

Exiting
 :w - write (save) the file, but don't exit
 :wq - write (save) and quit
 :q - quit (fails if anything has changed)
 :q! - quit and throw away changes

Search/Replace
 /pattern - search for pattern
 ?pattern - search backward for pattern
 n - repeat search in same direction
 N - repeat search in opposite direction
 :%s/old/new/g - replace all old with new throughout file
 :%s/old/new/gc - replace all old with new throughout file with confirmations

Working with multiple files

- :e filename - Edit a file in a new buffer
- :bnext (or :bn) - go to next buffer
- :bprev (of :bp) - go to previous buffer
- :bd - delete a buffer (close a file)
- :sp filename - Open a file in a new buffer and split window
- ctrl+ws - Split windows
- ctrl+ww - switch between windows
- ctrl+wq - Quit a window
- ctrl+wv - Split windows vertically