Global Information Assurance Certification Paper

Securing a Horde Webmail Server on RedHat 7.3 Step by Step

By James Philput
GIAC Certified UNIX Security Administrator
GCUX Practical - Option 1
Version 1.9
Date: 11/25/2003

Contents:

1. Executive Summary:

The purpose of this document is to detail the configuration and installation of a secure webmail server. The system is designed to allow company employees to send and receive email from anywhere that allows them access to the World Wide Web. As this server will have to be accessible to the Internet, certain steps must be taken to ensure it's security. This system, if compromised could allow a malicious user to gain access to company email, and possibly act as a jumping off point for the compromise of other systems on the company network. It is therefore imperative that this machine be installed securely, and maintained in that state once it is in production. This document will take you through the steps necessary to secure the server's operating system and services. Additionally, it will detail the steps necessary to maintain it in a secure state for the life of the server.

2. System Description:

2.1 Role:

This server is being built to address the growing need in this company for a remotely accessible email system. Although plans are in place to build a VPN that will allow remote access to the corporate network, both the Sales department, and the company's telecommuting workforce need a more immediate solution. This server will provide a web front end for employees to access their email from anywhere that they have access to the Internet. The server will connect to the primary mail server using the Horde webmail interface, and thus allow the users to send and receive email as though they were in the office. Additionally, we will be adding several modules from the Horde Framework to provide the users a few of the features that they have come to expect from their standard email clients. We will be adding Horde's Imp module to better facilitate the transfer of mail, Horde's Turba module to allow the users to store limited contact information in an online addressbook, and Horde's Kronolith module will allow remote users access to a Calendar/Scheduler from the web. The webmail server is designed to be a supplementary mail system, and not to be used as any users primary mail client. The heaviest use on this server will come from the company's 20 part time teleworkers. These individuals work from home for two to three days per week, and will use the Horde framework for email on those days.

2.2 Hardware:

The webmail server is being built on a moderately powerful computer. The server is running an AMD Athlon 2400+ with 256MB of RAM, and a 60GB hard drive. The server will be equipped with a 1.44MB floppy disk drive, a 48x CD-RW drive, and an integrated 32MB video card. A single 3Com 10/100 Ethernet

card will provide it's network connectivity. For this build we are using a lower powered hardware platform than is used on our other production servers. The reasons for this decision are 1, this server is intended to be in use for no more than three to six months, and 2, this server is not expected to be used by more than 60 users, only a portion of whom will be connected to it at the same time. If there is still a demonstrable need for this type of email access when, in 3 to 6 months, the company's VPN is up and running, this server will be rebuilt on a more powerful hardware platform.

2.3 Software:

This server will be built with the company's standard operating system, RedHat 7.3. We have found this to be a very stable operating system, with broad support from the Linux community. As with the operating system, all of the software used on this server is open source. Apache 1.3.29 was chosen for this project because it is widely supported, well documented and used on all company web servers. Version 1.3.28 was originally selected for this build, however a recent security vulnerability, a potential buffer overflow attack, caused this to be changed to Apache 1.3.29 which was released as a bug fix for the buffer overflow vulnerability in the previous version. (see http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-CAN-2003-0542) for full details on this vulnerability. No vulnerabilities have been published for this version so far. This Apache server will only accept SSL connections. To this end, Mod_SSL version 2.8.16-1.3.29 was selected to provide Apache with SSL support. The Horde Framework requires PHP in order to run. PHP version 4.3.3 will be used for this build. The Horde webmail framework was selected for this server because of its ease of use, and wide range of modules which can be installed to add greater functionality to the webmail server. For this build we will use Horde version 2.2.4. Horde is the main program in the Horde framework. It provides the basic scripting that its other modules work from. Imp version 3.2.2 is Horde's mail module, it provides the connection to a mail server, and the ability to send receive and compose email. Turba version 1.2.2, is Horde's addressbook module, it will be connected to a MySQL server, installed by RedHat 7.3. Kronolith 1.1 is Horde's calendar module; it will also use the MySQL database. Horde also needs the Pear program. Pear is installed as part of PHP 4.3.3, but we will need to install some extra modules to allow Horde to work correctly. The additional Pear packages needed by Horde are, Date 1.2, a package that allows date and time conversions between time zones, Log 1.7.0 a php logging system, HTML_Common 1.2.1, and HTML_Select1.2, which provide php with a means to better parse and display some of Horde's web pages. Lastly we will be partially installing a package called UW-IMAP version 4.1. The only aspect of the program that is needed are it's c-client libraries. These are required by Horde, and would not be installed otherwise.

Webmail Server at a Glance	
Hardware	**Software**
AMD Athlon 2400+	RedHat 7.3
256 MB RAM	Apache Web Server 1.3.29
60 GB Hard Drive	Mod_SSL 2.8.16-1.3.29
48x CD-RW	PHP 4.3.3
3.5" Floppy Drive	Horde 2.2.4
3COM 10/100 Ethernet Card	Imp 3.2.2
Integrated 32MB Video Card	Turba 1.2.2
	Kronolith 1.1
	UW-IMAP version 4.1
	Date 1.2
	Log 1.7.0
	HTML_Common 1.2.1
	HTML_Select1.2

2.4 Environment:

The corporate network is divided into two sections, the Production Network, and the Corporate LAN. The two sections are both on different NAT subnets. The production network is protected by an Internet connected firewall. The firewall is configured to forward the Production Network's External IP addresses to the appropriate server on the NAT subnet. For example, an incoming request to mail.thiscompany.com would arrive at the appropriate IP address on the firewall, and be forwarded to 10.10.10.2 on the Production Network. All of the external IP addresses must connect to the firewall before being forwarded to the appropriate server on the Production Network. In this manner, some of the company's servers can reside on the Production Network without being accessible from the Internet. A second firewall protects the Corporate LAN. The Corporate LAN firewall protects its network in the same manner that the Production Network firewall protects its network. The Corporate LAN firewall uses IP address translation to forward traffic from the Production Network to the appropriate machine on the Corporate LAN, and to forward traffic from the Corporate LAN to the appropriate machine on the Production Network. This provides an additional layer of protection to the Corporate LAN. The IP addresses for the Corporate LAN are configured using a different subnet than the Production Network, i.e. 10.10.0.0/24.

3. Risk Analysis:

There are two main areas of concern to look at when securing an Internet accessible Server, threats from inside the organization, and threats from outside the organization.

3.1 Internal:

An internal threat to the system could be one of several things. A disgruntled user could decide to try to gain access to the system in order to damage the file system or remove files. To prevent unauthorized use, access to this and every server will be limited solely to the personnel who need accounts on the machine in question. In the case of the webmail server, only system administrators will have accounts on the server. All users will be able to access webmail, but it will authenticate users by connecting to the mail server and authenticating them there. This setup also greatly reduces the chances that someone could accidentally delete or otherwise corrupt a needed file on the system. Another threat to the system is users gaining greater privileges on the system than they should by having physical access to the server. This is greatly reduced by having the production network machines at a different physical location than the users. The network operations center (NOC) and server room are in a separate building from the main company offices. Each person working in the NOC has an alarm code for building access. System administrators and the NOC manager have keys to the server room and the server cabinets. This reduction of physical access to the box greatly reduces the chances of an unauthorized user accessing the console and being able to elevate their privilege from there. The server will also be configured so that someone wanting to mount drives or other devices will not be able to unless they have root access. The user account restrictions and physical isolation of the server room from the users greatly reduce the prospect of an internal employee illegally elevating their privilege level, stealing information from the server, or adding malicious or trojaned code to the server.

3.2 External:

An External threat to the system would be a malicious user from the Internet. This user could be interested only in breaking in to prove that they can. They could be working for a competitor trying to steal company information, or they could be someone who is just interested in causing damage to any system that they can get into. The webmail server will not be storing, sending or receiving any mail itself. It will be connecting to the company mail server for that. In setting up the server so that it does not store mail or send mail, we have eliminated several potential problems before they could happen. The first is that a badly configured mail server can sometimes be used as a spam relay. Second, a user after company information, and hoping to read the mail on the system wouldn't find any. The malicious user would have to break into a second server for that information. The next major target service on this system is the web server. The web server would be a target just by being publicly accessible. The main concerns here are buffer overflow attacks, which could allow someone to gain access to the server, malicious users attempting to break in and deface the site, and someone trying to sniff username and password information as the user data travels across the Internet. We will try to address the buffer overflow attacks, and break in attempts via web exploits by keeping up with the latest bug fixes and security updates for the Apache web server, and all other services

accessible though it. The risk of user information being grabbed as it traverses the Internet can be greatly lessened with the use of SSL version 2, and only allowing the use of SSL's seven strongest Ciphers, which will encrypt all traffic between the client and the server. Additionally, the server will be configured only to use SSL encryption and an IPChains firewall will be installed on the server itself to only allow incoming connections on port 22 (SSH), and port 443 (SSL). Using this method all connections to this server will be encrypted. The Horde Framework has several known information gathering vulnerabilities that could be exploited by a malicious user to get information that might help in a system compromise. All of these vulnerabilities can be removed from the system by either turning on different options in the PHP configuration file, and/or removing test pages from Horde and its modules. The last major risks to this server are DoS, Denial of Service, and DDoS, Distributed Denial of Service attacks. Most of these attacks are automatically filtered by the Production Network firewall. However, there is very little that can be done if someone decides to launch a DDoS attack against the system. A DDoS attack tries to flood a networks available bandwidth and make it impossible for machines on that network to make a connection to the outside world. Many of these attacks make use of trojaned computers with broadband connections, making it relatively simple for a malicious user to expend many times more bandwidth to keep you from communicating than you can expend to communicate.

These precautions all reduce, but do not completely eliminate risks to the system. There is no way to make a usable, network connected server completely secure. Following the steps detailed below, and the recommendations for ongoing maintenance, should, however keep the server secure against most threats.

4. Step by Step Install:

This section details the actual software installation for the server. Some portions of the installation process have not been detailed completely as they do not have an effect on the security of the server. References will be provided in the text for those who want more detail on those portions of this document.

4.1 Operating System Install:

Note: Do **NOT** connect this machine to a network yet. It has not been hardened and therefore is not safe in a networked environment. Connecting the machine to the network is one of the last steps of this build.

For the installation we will be using store bought RedHat 7.3 installation media. RedHat 7.3 can also be downloaded from, http://ftp.redhat.com/pub/redhat/linux/7.3/en/iso/i386/. When downloading this or any other software from a public FTP server, be sure to verify the checksum of the downloaded file(s). There have been several instances in the past of

malicious users breaking into FTP sites, and replacing the available software with trojaned versions, which can allow unauthorized access to the system.

http://www.cert.org/advisories/CA-2002-30.html
http://www.cert.org/advisories/CA-1994-07.html
http://www.cert.org/advisories/CA-1994-14.html

Generally, an FTP site will have the correct checksums listed in a text file and available for download. On RedHat's server this file is called md5sum. You can find out the checksum of the downloaded files with this command:

```
md5sum <Filename>
```

Begin the installation by booting the server from the RedHat CD. Hit Enter when prompted and the install will begin. Follow the instructions on the screen, you will be asked to select your preferred language, keyboard type, and layout, and your mouse. After you finish on the Mouse Configuration screen, you will be asked for your install type. Select "Custom", this will allow you to make the necessary changes to the default install. Click "Next", and you will be taken to the Disk Partitioning Setup screen.

Select "Have the installer automatically partition for you"

This makes for an easier installation than using either Disk Druid, or FDISK.

Click Next

You should be taken to the Automatic Partitioning screen.

Select "Remove all partitions on this system"
Check "Review (Allows you to see and change the automatic partitioning results)"
Click Next

On the Disk setup screen you can make changes to the default partitions that the install program will create.

Select the line starting with /dev/hda1
Click the Edit button, and you should see a popup window. Set the values in that window as follows:

Mount Point = /boot
Filesystem Type = ext3
Size(MB): 50
Select "Fixed Size", and click OK.

Select the line starting with /dev/hda2
Click the Edit button, and you should see a popup window. Set the values in that window as follows:

Mount Point = /
Filesystem Type = ext3
Size(MB): 2048
Select "Fixed Size", and click OK.

Select the line starting with /dev/hda3
Click the Edit button, and you should see a popup window. Set the values in that window as follows:

Mount Point = swap
Filesystem Type = ext3
Size(MB): 512
Select "Fill all space to(MB): 1024", and click OK.

Click the "New" button, and set the popup values as follow:

Mount Point = /var
Filesystem Type = ext3
Size(MB): 5060
Select "Fixed Size", and click OK.

Click the "New" button, and set the popup values as follow:

Mount Point = /home
Filesystem Type = ext3
Size(MB): 5060
Select "Fixed Size", and click OK.

Mount Point = /usr
Filesystem Type = ext3
Size(MB): 5060
Select "Fixed Size", and click OK.

Click the "New" button, and set the popup values as follow:

Mount Point = /tmp
Filesystem Type = ext3
Size(MB): 5060
Select "Fixed Size", and click OK.

Click the "New" button, and set the popup values as follow:

Mount Point = /data
Filesystem Type = ext3
Size(MB): 30000
Select "Fill to maximum allowable size", and click OK.

RedHat's automatic partitioning program will resize the newly created partitions to make better use of disk space. The final partition table will look like this:

|New| |Edit| |Delete| |Reset| |Make RAID|

Device	Start	End	Size(MB)	Type	Mount Point	Format
/dev/hda						
/dev/hda1	1	6	47	ext3	/boot	yes
/dev/hda2	7	4382	34326	ext3	/data	yes
/dev/hda3	4383	5027	5060	ext3	/tmp	yes
/dev/hda4	5028	7297	17806	extended		
/dev/hda5	5028	5672	5060	ext3	/usr	yes
/dev/hda6	5673	6317	5060	ext3	/home	yes
/dev/hda7	6318	6962	5060	ext3	/var	yes
/dev/hda8	6963	7223	2047	ext3	/home	yes
/dev/hda9	7224	7297	580	swap		yes

These changes will allow you to mount the individual filesystems with different permissions, such as, read only, nosuid, and nodev; all of which will be gone over in the /etc/fstab section. Having the ability to mount different filesystems with different permissions can greatly increase the security of the server.
Click Next, and a window will appear that reads:
:

WARNING!! WARNING!!
You Have Selected to Remove all Partitions(ALL DATA) on the Following Drives:
/dev/hda
Are you sure you want to do this?

Agree, and continue to the Boot Loader Information Screen, under the Boot Loader Configuration screen, select Use Grub as the boot loader. Proceed to the next screen, and check the box labeled "Use a Boot Loader Password", and enter a password. This , along with disk mounting option we will set later,will force a password to be entered on boot before the machine will come up. Since no one but an authorized user should reboot the server, the password can keep an unauthorized user from being able to pass new options to the kernel on reboot. In the "Install Boot Loader on:" section,

select /dev/hda Master Boot Record

Under Partition: /dev/hda2

set the Type to ext3.

Select, Default Boot image.

On the Boot Label line, enter the following:

Webmail Server

Click Next

On the Firewall Configuration screen,

Select "High" under Security level

Select "Customize firewall rules"

Check SSH, and WWW

These settings start the build off with only two ports listening for connections, port 22 (SSH) and port 80 (HTTP). This means that if the machine were connected to the network right now it would only accept connections on those two ports, and automatically reject connections to any other ports. Click Next, and you will be taken through the screens for Additional Language selection, and Time Zone selection. On the Account Configuration Screen, enter the system root password. According to the local password policy, this must be a minimum of nine characters long, and consist of letters in upper case and lower case, numbers and symbols. Setting up the password in this way makes it much more difficult for a malicious user to get the password by either exhaustive guessing, or a brute force attempt to crack it. Once you have set the root password according to policy, add at least one additional account so that you will not have to log in as the root user. Click Next. Change nothing under Authentication Information, click Next. Under Package Group Selection de-select all, click Select individual packages, and then click Next. On the Individual Package selection screen select only the following packages:

dump
rmt
mysql
mysql-devel
mysql-server
mysqlclient9
vim-enhanced
nc
openssh
openssh-server
openssh-clients

bind-utils
gnupg
sudo
tripwire
cpp
gcc
gcc-c++
perl
perl-CGI
perl-CPAN
perl-DB-file
perl-NDBM-File
automake
automake15
binutils
gettext
glibc-utils
libtool
libtool-libs13
make
kernel-utils
python
freetype
gd
libjpeg
libpng
autoconf
m4
libtool-libs
glibc-devel
ucd-snmp
krbafs
libstdc++-devel
gmp
glibc-kernheaders
flex
pam-devel
mm

Click Next

On the About to Install screen, click Next.

On the Boot Disc Creation screen, click skip boot disk creation, we will not need a boot floppy for this machine. Click Next.

At this point the installer will begin to format the hard drive; it will then install the operating system. You will be prompted to change CDs during the install. When the install is complete click exit. The system will reboot.

4.1.1 Patches and Updates:

For this step you will need to be at an Internet connected computer with a CD burner. Before continuing work on the new server we need to get the most recent updates for the operating system. Use an FTP client to go to the following addresses

updates.redhat.com/7.3/en/os/i386
updates.redhat.com/7.3/en/os/i686
updates.redhat.com/7.3/en/os/noarch

You will need to download all available updates to a single directory. This will take a little while as the updates total about 550 megabytes worth of data. Before you burn the files to CD, go into the download directory and remove the files, kernel-bigmem, since the server does not have 4 or more gigabytes of RAM, and kernel-smp, since this is not a multi-processor machine. The files from the i686 directory are actually replacements for their i386 counterparts, so it is safe to delete

glibc-2.2.5-43.7.i386.rpm
glibc-2.2.5-43.7.i386.rpm
kernel-2.4.20-20.7.i386.rpm
openssl-0.9.6b-35.7.i386.rpm

since they have i686 versions. Burn the downloaded files to CD and move them to the server. Log into the server as root and type the following commands:

```
mount /dev/cdrom
rpm -Fv  /mnt/cdrom/updatedirectory/*.rpm
```

The command rpm -Fv will only update or "Freshen" a package if an older version of that package is installed on the system. The v causes verbose information to be sent to the screen. This command will update all installed packages to newer versions, if available. The kernel updates are included. Normally a kernel update should be installed using the -i option instead of -F. The -i would install the module, and you would be able to rollback to the earlier kernel if there were problems with the new one. In this case, the kernel update has already been tested and used at the site, so for ease of install, we did not separate the kernel update from the rest of them.

4.2 System Configuration:

This section details how to turn off unnecessary services, set runlevels, and tighten up the machine in general. The chkconfig command is used to see what

services are running on a machine, and at what runlevel they are operating on. It can also be used to turn off unneeded services. Run the following command to see what is currently running on the system.

```
chkconfig --list
```

This will give a list of the installed services, and their runlevels. Run the following commands to turn off unneeded services.

```
chkconfig --level 2345 apmd off
chkconfig --level 2345 gpm off
chkconfig --level 2345 sendmail off
chkconfig --level 2345 kudzu off
chkconfig --level 2345 keytable off
chkconfig --level 2345 atd off
chkconfig --level 2345 microcode_ctl off
chkconfig --level 2345 netfs off
chkconfig --level 2345 rawdevices off
chkconfig --level 2345 anacron off
chkconfig --level 2345 snmpd off
chkconfig --level 2345 snmptrapd off
chkconfig --level 2345 smartd off
```

Next we will set the default umask for the system. " Before the mask is applied, a directory has permissions *777* or *rwxrwxrwx*, a plain file *666* or *rw-rw-rw-*. The *umask* value is subtracted from these default permissions after the function has created the new file or directory. Thus, a directory will have permissions of *775* by default, a file *664*, if the mask value is *(0)002*"[1].

```
vi /etc/rc.d/init.d/functions
                set the default umask to 022
:wq
```

This umask setting sets the default file creation permissions to 755, which allows the file owner to read write and execute files, but limits everyone else to read and execute. Next, edit the inittab file.

```
cd /etc
vi inittab
        set the default Run Level to 3 as in:
                        id:3:initdefault:
```

This sets the system to a default full multiuser mode, which allows remote access to the machine. This is necessary for remote administration. From a security

[1]Garrels, http://www.tldp.org/LDP/intro-linux/html/sect_03_04.html#sect_03_04_02_02

stand point the server would be more secure at run level 1, single user mode, but that would not make for a very useable server.

remove the line cs::ctrlaltdel:/sbin/shutdown -t3 -r now

Removing this line disables the use of ctrl-alt-del to reboot the server. We do not want anyone to be able to reboot the server without logging in, and having the proper permissions.

comment the lines
```
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6
```

Commenting out the mingetty lines disables mingetty on everything but the console. This can prevent login via serial port if someone has physical access to the box, and tries to get in via that route. This step also disables console access on all but tty1, which can reduce the possibility of an admin forgetting to logout of a privileged session, when logged into the console on more than one tty.

```
x:5:respawn:/etc/X11/prefadm -nodaemon
```

Since this machine is not running XWindows, we do not need this line.

```
:wq
```

Now we need to edit the sysctl.conf file. Here we can set some of the kernel options that will make the server both more stable, and more secure.

```
vi sysctl.conf
add the line                net.ipv4.tcp_max_syn_backlog=4096
```

In this step we have increased the default number of connections that the server can store without them having been answered from 1024 to 4096. This is done as a small measure of protection against SYN floods. A SYN Flood occurs when an attacker sends numerous SYN packets to a machine from a spoofed IP address, falsely trying to initiate a connection. "A SYN flood causes so many TCP/IP open sessions that the system becomes overwhelmed and cannot handle any more network traffic"[2]. Further information on SYN Flood attacks can be found at http://www.cert.org/advisories/CA-1996-21.html.

In this file we can make changes to the mountable devices on the system.

[2]Network World Fusion, http://www.nwfusion.com/links/Encyclopedia/S/674.html

vi fstab
add nosuid, nodev, and noauto to the /dev/cdrom, and /dev/floppy lines
set /data to rw,nosuid,nodev
set /tmp to nosuid,nodev
set /usr to ro,nodev
set /var to rw,nosuid,nodev
set /home to rw,nosuid,nodev
:wq

The nosuid command makes a device or filesystem unable to run any program that needs to reset it's UID. This means that a program cannot use that command to change its permissions. The reason that /usr cannot use nosuid is that the sudo command must be able to set UID to root. The nodev command makes it impossible to mount other devices in that filesystem. The noauto command keeps the floppy and CD drives from being automatically mounted when a disk is loaded. We have setup the filesystem in such a way that only the root user can mount the floppy or the CD drive. The /usr filesystem is mounted in read only mode, and no devices can be created on it. Both /var, and /home are mounted in read and write mode, but no program on either filesystem can set uid to root, and no one can mount a new device on either.

vi syslog.conf
Add the following line
. @192.168.0.7[3]
:wq

This line sets the syslog daemon to log to the remote syslog server. We did this because it makes it much more difficult for a user who has broken into the server to modify the logs. The malicious user would have to break into a second server before being able to change the log files.

The login.defs file contains the password settings. Here we can change the expiration time, and the minimum length of a users' password.

vi login.defs
set PASS_MAX_DAYS 30
PASS_MIN_LEN 9
uncomment USERDEL_CMD

The commands above set the minimum length of all passwords to 9 characters, set the expiration on all passwords to 30 days, and enable the userdel command.

[3] Hines, http://www.linuxsecurity.com/feature_stories/remote_logserver-3.html

The userdel command allows for easier removal of accounts and is enabled only for easier account management.

```
cd /etc/ssh
vi sshd_config
```

Uncomment and change lines as follows:

```
Port 22
Protocol 2
PermitRootLogin no
PasswordAuthentication yes
PermitEmptyPasswords no
Banner /etc/issue
```

Comment out ALL x11 lines

```
:wq
```

The changes to the sshd config file force the ssh daemon to only listen on port 22, only accept ssh version 2 connections, disallow root login via ssh, disallow any logins with empty passwords, and configures sshd to display a custom banner when an ssh client connects. Since we have setup the server only to allow ssh connection attempts to port 22 uncommenting the port line is only good for redundancy. Forcing ssh version 2 is useful because there are some known bugs in version 1. Full details on these vulnerabilities can be found at:

http://www.kb.cert.org/vuls/id/945216
http://www.kb.cert.org/vuls/id/25309
http://www.kb.cert.org/vuls/id/161576
http://www.kb.cert.org/vuls/id/13877

Keeping root from logging in remotely is a very good idea security wise. If you have several people on your site with the root password, and someone who shouldn't have it manages to get it, you would be unable to track who was actually logging in. Also, by disallowing empty passwords we further reduce the chances that someone could log in using a daemon account, or create and use an account with no password set. Lastly, by setting the banner line, you can configure the default banner to something other than the SSH version, and add a security notice.

```
cd ../security
```

In this file we can make changes to prevent core dumps by users on the system.

vi limits.conf

uncomment the line

* soft core 0

Change the newly uncommented line to

* hard core 0

delete all other lines
:wq

We have uncommented the line that sets the maximum size of a core file and set it to 0. This means that no core file will be created if there is a core dump. Further, we have changed the setting from soft to hard. This will prevent a malicious user from being able to change this setting without being the root user.

The console.perms file defines the default device permissions given to all users logged in from the console.

vi console.perms
comment out ALL floppy CD and ALL unused devices
:wq

By commenting out the floppy and CD drives, and removing all of the other drives, we can prevent unprivileged users from attaching and mounting new devices, or mounting existing devices. This could prevent someone from connecting a new device to the server and transferring data to or from it. Likewise, it prevents a non privileged user from transferring data to or from an existing device.

cd ..

4.3 System Account Removal:

We can now begin to remove unneeded system accounts. This makes it much more difficult to start a new system service that was not installed by default. It will also prevent a malicious user from using them as backdoor accounts onto the system.

more /etc/passwd

Make a list of unused accounts to remove

q

NOTE: Since the userdel command tries to remove a users home directory when it is run, you will get an error after removing each of the following system accounts. This is normal and occurs because these are system accounts without home directories.

```
userdel     adm
   "        lp
   "        mailnull
   "        news
   "        gopher
   "        ftp
   "        uucp
   "        mail
   "        operator
   "        games
```

At this point we can add the user accounts to the system. The Horde webmail software pulls its authentication from the mail server it connects to. Since this installation will connect to a remote server for mail, we do not need to install user accounts on this server for the mail users. Only system administrators should have accounts on this server.

As root, use the useradd and passwd commands to add the necessary users.

```
useradd     sysadmin1
useradd     sysadmin2
useradd     sysadmin3
```

Setup the passwords for each account according to the password policy, and have each of the sysadmins change the password the first time they login.

```
passwd      sysadmin1
Changing password for user sysadmin1.
New password: <Enter Temporary Password>
Retype new password: <Enter Temporary Password>
passwd: all authentication tokens updated successfully.
```

Repeat this step with each account until all user passwords are set.

Now that the user accounts have been entered, we should setup access control for each user. At this site two of the 4 system administrators have full root access to the systems. The 2 junior administrators have more restricted access. On this server we will be using the sudo program to allow the users to run some commands as root. The sudo program allows a user to upgrade his or her privilege to that of root for certain commands. More information and

documentation on the sudo program can be found at http://www.courtesan.com/sudo/ The user types in,

```
sudo <command>
```

The system then prompts for the user's password, and when it is entered correctly, the command is executed. The sudo user may then enter other commands to which they have access in the /etc/sudoers file. If the user has not entered a command in five minutes, the system will prompt for a password before another sudo command will be executed. Sudo helps to limit users to commands that they need to use on a day to day basis. This makes it unnecessary to give the root password to every user that needs to run some commands that require root privilege.
To setup sudo access you will need to edit the /etc/sudoers file with the visudo command. Visudo locks the file to be edited and runs syntax checks on the file before allowing it to be closed.

```
visudo
```

Add the following lines

```
sysadmin      ALL = /bin/su -
sysadmin1     ALL = /bin/su -
sysadmin2     ALL = /sbin/service mysqld restart,
/usr/local/apache/bin/apachectl restart, /bin/kill
sysadmin3     ALL = /sbin/service mysqld restart,
/usr/local/apache/bin/apachectl restart, /bin/kill
:wq
```

In this configuration The two system administrators who have root access may use sudo to get full root access. The two junior system administrators can restart the Apache, and MySQL servers, and kill processes.

4.4 Banner Setup:

At this point we can setup the system login banners. These will be the messages that a user sees when logging in via ssh or the console.

```
vi issue
```

Add the following:

```
WARNING Unauthorized uses are prohibited, and will be prosecuted. All
                Connections are logged and traced.
:wq
```

```
vi motd
```

Add the following:

--WARNING-- Unauthorized uses are prohibited, and will be prosecuted. All Connections are logged and traced.

:wq

4.5 Network setup:

Now that the OS has been hardened we are ready to configure the computer for Network access.

```
cd /etc/sysconfig/network-scripts
vi ifcfg-eth0
```

Set the following lines to the values below

```
BOOT PROTO = static
IPADDR = 192.168.0.4
NETMASK = 255.255.255.0
GATEWAY = 192.168.0.1
```

:wq

The OS is hardened and the server is configured for network access. Since some of the files that have been modified will not be effective until a system restart, restart the server, but do not connect it to the network.

4.6 Server software installation:

Once you have logged back into the server, begin the additional installation. You will need to download Apache, PHP, MOD_SSL, UW-IMAP, Horde, Imp, Turba, and Kronolith. As we did in the OS upgrade section, use another machine to download the files, and burn them to cd. Once that is done mount the cd, and copy the files to their install directories from there. Make sure that the versions you download are the same as those listed at the beginning of this document, or some of the instructions may fail.

First we need to install the UW-IMAP software. UW-IMAP is needed by Horde for its c-client libraries. Since these libraries are the only things needed by the system, we will not be doing a complete installation of this package.

4.6.1 Install UW-IMAP:

Copy the source tarball to the /usr/local source directory and unpack it.

```
cp imap.tar.Z /usr/local/src/
tar zxpf imap.tar.z
```

This should create a new directory called imap2002e.

```
cd imap2002e
```

Run the following commands to setup UW-IMAP for Horde.

```
./configure
make lrh
```

This command sets up the software for installation on a RedHat OS.

```
ln –s c-client include
mkdir lib
cd lib
ln –s ../c-client/c-client.a
```

Now that UW-IMAP is setup, we can install Apache, PHP, and MOD_SSL.

NOTE: the order of installation for Apache PHP and MOD_SSL is important, and the software may not run unless installed in the proper order.

4.6.2 Install Apache, PHP, and Mod_SSL:

Copy the source tarballs to /usr/local/src

```
cp apache_1.3.28.tar.gz /usr/local/src
cp php-4.3.3.tar.gz /usr/local/src
cp mod_ssl-2.8.15-1.3.28.tar.gz
```

Unpack the files.

```
tar zxpf apache_1.3.28.tar.gz
tar zxpf php-4.3.3.tar.gz
tar zxpf mod_ssl-2.8.15-1.3.28.tar.gz
```

First we will need to apply Mod_SSL to Apache's source tree.

```
cd /usr/local/src/mod_ssl-2.8.15-1.3.28
./configure --with-apache=/usr/local/src/apache-1.3.28
```

Next we need to run a basic setup of Apache for PHP to use in its installation.

```
cd /usr/local/src/apache-1.3.28
./configure --enable-module=rewrite --enable-shared=rewrite
make
make install
```

Now that the basics are taken care of, setup PHP.

```
cd /usr/local/src/php-4.3.3
./configure    --with-apache=/usr/local/src/apache-1.3.28
               --with-mysql
               --with-gettext=/bin/gettext
               --with-openssl
               --enable-sockets
               --with-imap=/usr/local/src/imap2002e
               --with-pear
               --with-zlib

make
make install
```

- Install Apache

```
cd /usr/local/src/apache-1.3.28
SSL_BASE=SYSTEM EAPI_MM=SYSTEM ./configure
                    --enable-module=rewrite
                    --enable-shared=rewrite
                    --enable-module=proxy
                    --enable-shared=proxy
                    --activate-module=src/modules/php4/libphp4.a
                    --enable-module=ssl
                    --enable-shared=ssl
make
make certificate
```

The make certificate command generates a signed ssl server certificate for the web server. For this installation, we will be using a self generated, and signed key. This is a server certificate which is not recognized as valid on the Internet in general. To get a valid server certificate you would need to contact one of the Internet Certificate authorities CA, and pay them to generate and sign on for you. Since this server is intended for internal use only we are safe using a self signed key. Once you have entered the make certificate command you will be prompted for information about your site and location. Hit enter for the default answer to each question, enter, and confirm your passphrase, and the certificate will be generated automatically. See Appendix A.

```
make install
```

4.6.3 Horde Configuration:

Before Horde can work correctly we will need to install several additional pear modules that this setup of Horde requires in order to run. Pear packages can be

downloaded from http://pear.php.net. For this installation we will need to download Date 1.2, a package that allows date and time conversions between time zones, Log 1.7.0 a php logging system, HTML_Common 1.2.1, and HTML_Select1.2, which provide php with a means to better parse and display some of Horde's web pages. Full descriptions of each package can be found at http://pear.php.net/packages.php. Once the packages are downloaded to the server, run the following command.

```
pear install <packagename.tgz>
```

Pear is now setup correctly for Horde. Begin the installation by unpacking Horde, Imp, Kronolith, and Turba. Once that is done, copy the new directories to their permanent location. Copying the directories in this manner leaves a clean copy of the original directories

```
cp -R horde-2.2.3 /path/to/document/root/horde
cp -R imp-3.2.1 /path/to/document/root/horde/imp
cp -R turba-1.2 /path/to/document/root/horde/turba
cp -R kronolith-1.1 /path/to/document/root/horde/kronolith
```

Once you have done that you will need to prepare Horde's configuration files. The files we need are in the config subdirectory of Horde, Imp, Turba, and Kronolith. All of the files in those directories end in .dist. The .dist needs to be removed from the end of each file name.

```
cd /path/to/document/root/horde/config
cp pref.php.dist prefs.php
```

Repeat the copy command for each file in the directory. Repeat this process in the config directories for each of the installed Horde modules.

4.6.4 Horde Module Configuration:

Before you begin, make certain that MySQL is running. This is done by running the command.

```
service mysqld start
```

Once MySQL is running, we will need to setup a root password for MySQL. We will use the following commands to set the password.

```
mysql -u root
set password for 'root' = password('<root'spassword>');
quit
```

Log in to verify the password change.

```
mysql -u root -p
quit
```

Now that the root password is set, we can begin to configure Horde, and its modules for use. The Horde setup contains very little information that affects the security setup of this machine. Therefore, only limited explanatory detail is supplied for this section. Detailed explanations of the function of each of the modified files, and example scripts can be found in the Horde Administrator's FAQ at http://www.horde.org/faq/admin/print.php.

4.6.5 Horde:

First we will setup the MySQL scripts the Horde will use to make its database and tables in MySQL.

```
cd /path/to/document/root/horde/scripts/db.
```

Add the line USE horde; to line 3 of auth.sql, category_mysql.sql, prefs.sql, and vfs.sql

Run the following commands:

```
mysql -u root -p <mysql_create.sql
mysql -u root -p <auth.sql
mysql -u root -p <category_mysql.sql
mysql -u root -p <prefs.sql
mysql -u root -p <vfs.sql
```

Now we will begin to setup Horde's files for our server. Descriptions of each file, and line change can be found in the comments of each file.

```
cd /horde/config
vi horde.php
```

Set the following options:

```
Line 162 = $conf ['prefs']['driver'] = 'sql';
Comment out line 166
Uncomment lines 171 to 176
Line 174 = Change the password to horde
Comment out lines 236, and 243
Line 269 = $conf ['sessionhandler']['type'] = '';
:wq

vi registry.php
```

Set the following options:

```
Uncomment lines 23, 24, and 36 to 40
Line 116 = 'status' => 'active'
```

Line 125 = 'status' => 'active'
Line 134 = 'status' => 'active'
:wq

4.6.6 Imp:

cd /horde/imp/config
vi conf.php
Set the following options:
Line 37 = $conf ['menu']['apps'] = array('imp','turba','kronolith','horde');
:wq

vi servers.php
Set the following options:

Disable all of the example servers but the one you want to use. This can be done by adding an _ to the top line of the server's array as in:

```
$servers ['pop3'] = array (
        should be
$servers ['_pop3'] = array (
```

Once you have done that, set up the server with the correct information. Setup an IMAP server connection as follows:

```
$servers ['imap'] = array (
        'name' => 'Webmail',
        'server' => 'webmail.localhost.localdomain',
        'protocol' => 'imap/notls',
        'port' => 143,
        'folders' => '',
        'namespace' => '',
        'maildomain' => 'webmail.localhost.localdomain',
        'smtphost' => 'webmail.localhost.localdomain',
        'realm' => '',
        'preferred' => '',
);
```

:wq

4.6.7 Turba:

cd /horde/turba/config
vi conf.php
Set the following options:
Line 32 = $conf ['menu']['apps'] = array('imp','turba','kronolith','horde');

:wq

vi sources.php

Comment out all of the example databases but the one that will be used. For this server we will only be using the prefs based Private Addressbook, which was setup earlier in the horde module section.

:wq

cd /horde/turba/scripts

Run the following command:

mysql -u root -p <mysql_create.sql

4.6.8 Kronolith:

cd /horde/kronolith/config
vi conf.php

Set the following options:

Line 22 = $conf ['calendar']['driver'] = 'sql';
Comment out lines 28, 32, 33
Uncomment lines 36 to 41
Line 39 = $conf ['calendar']['password'] = 'horde';
Line 66 = $conf ['menu']['apps'] = array('imp','turba','kronolith','horde');
:wq

cd /horde/kronolith/scripts
vi kronolith.sql

Add the line USE horde; to line 3.

:wq

Run the following command:

mysql -u root -p <kronolith.sql

Once all of the changes are made and the scripts are run, we need to change the horde password in MySQL to the one that was entered in the configuration files above.
Enter the following commands.

mysql -u root -p

Enter the MySQL root password and type in the following command.

set password for 'horde' = password('<horde'spassword>');

quit

4.7 Web Server Configuration:

Now that the Horde is configured and installed, we need to make changes to the httpd.conf file. This is the file that Apache pulls its information on what information to serve, and from where.

```
cd  /path/to/apache/conf
vi httpd.conf
```

Add the following line to the AddType section.

```
AddType application/x-httpd-php .php
```

In the mod_alias section add the following lines

```
Alias /path/to/document/root/horde/ "/horde/"

<Directory "/path/to/document/root/horde">
        Options Indexes Multiviews
        AllowOverride None
        Order allow,deny
        Allow from all
</Directory>

Alias /path/to/document/root/horde/imp/ "/horde/imp/"

<Directory "/path/to/document/root/horde/imp">
        Options Indexes Multiviews
        AllowOverride None
        Order allow,deny
        Allow from all
</Directory>

Alias /path/to/document/root/horde/turba/ "/horde/turba/"

<Directory "/path/to/document/root /horde/turba">
        Options Indexes Multiviews
        AllowOverride None
        Order allow,deny
        Allow from all
</Directory>

Alias /path/to/document/root/horde/kronolith/ "/horde/kronolith/"

<Directory "/path/to/document/root/horde/kronolith">
        Options Indexes Multiviews
        AllowOverride None
```

```
        Order allow,deny
        Allow from all
    </Directory>
```

Uncomment the line, “NameVirtualHost *”.

In the SSL Virtual Hosts section, add the following lines:

```
    <VirtualHost *:80>
        DocumentRoot /horde/imp/
        ServerName internal.sans.org
        DirectoryIndex index.php
        ErrorLog logs/horde-access_log
        CustomLog logs/horde-access_log common
    </VirtualHost>
```

:wq

Now the apache web server is completely configured, and we should be able to start it with no errors. Run the following command.

```
/usr/local/apache/bin/apachectl startssl
```

This should result in a message that says:

```
/usr/local/apache/bin/apachectl start: httpd started
```

The server software installation is nearly complete. Now we will begin making changes to secure the new software that we have installed.

4.8 Securing the server Software:

This section deals with the final modifications to the server before it is ready to be placed on the network, and made accessible from the Internet. When finding out what needs to be secured on a new box several websites are invaluable tools. http://www.cert.org, http://www.cve.mitre.org/ , and http://www.securityfocus.com all have searchable databases of vulnerabilities which can tell you if a given piece of software a. has any currently known security problems, and b. how to fix them. Security focus also has a searchable database of the Bugtraq mailing list. Additionally, http://www.apacheweek.com/features/security-13 has a list of known vulnerabilities in the Apache web server, along with a list telling you which release of Apache the bug was fixed in. http://www.modssl.org/docs/2.8/ssl_howto.html has an excellent manual that details setting the Apache web server up for strong encryption.

4.8.1 Apache 1.3.29:

There are not currently any known vulnerabilities with this release of Apache. This does not mean that this release does not have any vulnerabilities, it just means that they have not been found and published yet.

4.8.2 PHP 4.3.3:

I was unable to find known vulnerabilities for this release of PHP, however making modifications to the php.ini file can help to improve security. PHP comes with two versions of the php.ini file by default, one is standard, and allows easy php scripting, the other is recommended by php due to its security fixes. We will use that one. Run the following commands to setup the recommended php.ini.

```
cd /usr/local/src/php-4.4.3
cp php.ini-recommended /usr/local/lib/php.ini
```

This version of the php.ini file sets the following security related variables:

register_globals = Off

"When on, register_globals will inject (poison) your scripts will all sorts of variables, like request variables from html forms"[4]. This turns off global variables in scripts, and forces each script to use explicitly defined variables.

display_errors = Off

This turns off logging errors to the screen, which can give useful information to a malicious user.

log_errors = On

This turns logging on. All php errors are logged but not displayed to the screen as part of a web page.

There is one other change to be made to this file. We need to set the session.use_only_cookies line to on. This makes it much more difficult for a malicious user to hijack a user's session by using the user's session ID. A full description of this vulnerability can be found at http://www.securityfocus.com/bid/8399/info/.

```
vi /usr/local/lib/php.ini
```

Change the line reading

```
session.use_only_cookies = 0
```

to

```
session.use_only_cookies = 1
:wq
```

[4] PHP.net, http://www.php.net/manual/en/security.registerglobals.php

4.8.3 Mod_SSL:

I was not able to find any vulnerabilities listed for the version of Mod_SSL installed on this system. However, the Howto document referenced above gives instructions for disabling weak encryption in SSL.

In this section we will be setting up Mod_SSL and Apache to use only SSL version 2 and the seven strongest of it's ciphers. To do this, we will need to edit the httpd.conf file.

```
cd /path/to/apache/conf
vi httpd.conf
```

Add the line

```
SSLProtocol -all +SSLv2
```

[5]

on the line directly above the SSLCiphersuite line. Change the SSLCiphersuite line as follows.

```
SSLCipherSuite SSLv2:+HIGH:+MEDIUM
```

[6]

```
:wq
```

Restart the apache server with the command,

```
/path/to/apachectl restart
```

4.8.4 Horde Imp Turba and Kronolith:

The Horde suite of applications is the least secure of the programs installed on this server. As mentioned above, a line of the php.ini file was changed to fix a vulnerability to do with potential session hijacking. That vulnerability should have been fixed in the version of Horde that we are using, but it is better to be safe than sorry. The first Horde framework specific vulnerability that we will be fixing is in both the Turba, and Kronolith modules. A call to the status.php file can cause an error which would display the full path of the file. This is not a critical vulnerability, but it could give a malicious user information about the server's directory structure. Details on this vulnerability can be found at http://www.securityfocus.com/bid/7622. The status.php file is not needed on a production machine, as it is only used for checking the working status of the modules. The best way to solve this problem is to delete the file.

```
cd /path/to/turba
rm status.php
```

Answer yes when prompted to confirm the deletion, and repeat the above process to remove Kronolith's status.php file.

[5] Engelschall, http://www.modssl.org/docs/2.8/ssl_howto.html#ToC2

[6] Engelschall, http://www.modssl.org/docs/2.8/ssl_howto.html#ToC2

I was unable to find any specific vulnerability references for the next vulnerability. As was the case with Kronolith and Turba this is also an information gathering vulnerability. The test.php files in both Horde, and Imp give out a great deal of information about the server. These pages, when Internet accessible, give the web server version, php version, version numbers of all Horde framework applications, kernel version, and several other useful pieces of information. Again the best way to fix this is to delete the test.php files. These files should only ever be used during the initial setup of the server and are not used at any other time.

```
cd /path/to/horde
rm test.php
```

Answer yes when prompted to confirm the deletion, and use the same steps to remove Imp's test.php file.

4.9 Tripwire Setup:

The last stage of the installation is the Tripwire setup. Tripwire is a file integrity checker. It can be setup to run via a Cron job, and when run, it will check the files on your system against a database it has generated and notify you of any changes. We saved this for last because it would have given error reports frequently if we had set it up earlier in the build. This way we can start it on a system that should not change much, so we should only get errors we want to see as opposed to those that would have been caused by this installation. Tripwire itself was installed during the main OS install. All that we have to do now is set it up.

The first step in setting up Tripwire is running its setup script. The script will setup the local and site keys, which are used for encrypting and decrypting various Tripwire files, and generate the initial policy file. The policy file is where Tripwire stores its list of files to check, and what level of severity a change to each one is.

```
cd /etc/tripwire
```

run the setup script.

```
./twinstall.sh
```

You will be prompted to enter and confirm the passphrase for both your local key and your site key. Be sure to follow the password policy when creating the passphrases.

Once this is done we need to setup the tripwire policy file. We will need to edit the policy file to remove files that either don't exist on the system, or that we want monitor in addition to the defaults already in the file.

```
vi twpol.txt
```

First, we need to setup Tripwire to send an email alert when it detects a problem. This is done by adding an emailto: line to each of the rules in the policy file. Add a comma to the end of the "Severity" line of each rule. Then, immediately below the "severity" line add another line reading emailto = administrator@localhost.localdomain, substituting the appropriate address for administrator@localhot.localdomain. See the example below.

```
(
        rulename = "Critical configuration files",
        severity = $(SIG_HI),
        emailto = administrator@localhost.localdomain
)
```

After adding the emailto lines, we will need to comment out or delete all unnecessary lines from the file.

Once the unneeded lines have been removed or commented out, we can start Tripwire. The first step in getting Tripwire running with it's new policy file is to run the Tripwire database initialization command.

```
tripwire -m i
```

The -m i tells Tripwire that it is running in database initialization mode. This command will generate some errors, because it is still using its default policy file. The next command:

```
tripwire -m p twpol.txt
```

tells tripwire to go into policy update mode, and use the twpol.txt file as its input for the new policy. Tripwire should now be setup with the modified policy file, and ready to run. As the final check, we need to run,

```
tripwire -m c
```

this runs Tripwire in check mode, the same way it will be run automatically.

Now we will verify that Tripwire is setup as a cron job. The Tripwire installation should have placed a script called tripwire-check into the /etc/cron.daily directory. Verify that it is there.

```
cd /etc/cron.daily
ls
```

Tripwire will run once a day, and error reports will be emailed to the address specified in the policy file. The server installation is now complete. Connect the server to the network, and type the command

ifup eth0

This should bring up the network interface. We are now ready to test the build.

5. Ongoing Maintenance:

Now that the server is built and secured, we need to ensure that it will stay secure. One of the first steps in keeping it that way is creating a policy that defines what checks need to be made on the system, and when they need to be made. One of the best things that can be done to keep a system secure is to stay on top of its software updates. The system administrators at this site are all subscribed to the RedHat Watch mailing list. http://www.redhat.com/mailman/listinfo/redhat-watch-list/ This list provides up to date information on new security patches and bug fixes released by RedHat. The administrators also subscribe to the Bugtraq mailing list. Bugtraq is an email list to which people send potential security problems and bugs for review. Bugtraq is an excellent source of information on new software problems and vulnerabilities. Keeping up to date on these two lists should provide the ability to patch security vulnerabilities quickly after they are found. The site maintenance policy also dictates a twice weekly review of all Tripwire reports from the servers. This review helps to verify that Tripwire is running properly, and at the correct times. The webmail server sends its logs to a remote Syslog server. If a malicious user were to break into the webmail server, this configuration would make it much more difficult for that user to hide the traces of a break in by modifying the server logs. The malicious user would have to break into a second server to modify the logs. This method of logging helps to ensure a valid audit trail in the event of a system compromise. The Syslog server is running Logsurfer, a log monitoring, and alert tool that is configured to email any logs which it's configuration file says are problems to the system administrators. These problems include, but are not limited to, failed logins, misconfigured packets, and information on traffic blocked by the firewall. The security policy also dictates that the testing tool, Nessus be run on all systems once every two weeks. Nessus uses NMAP to scan each system on the network for open ports, and then uses a collection of modules, to attempt exploits against each system. The modules are scripts containing the code for a given exploit, and can be downloaded from http//:www.nessus.org. Once it has finished its run Nessus generates a report on each machine it has tested. The report lists the security problems, if any that it has found. The weekly Nessus reports are then gone through, and any problems that are found are solved, either by upgrading the software on the machine to a new version, if available, or by making changes to the system configuration so that the exploit can no longer work. With the policy of ongoing updates, log monitoring, review, and network scanning, the system administrators at this site should be able to maintain, and even increase the security of this system.

6. Configuration Check:

Now that the installation is complete, we will check it to make sure that the server is both working, and secure.

6.1 Webmail:

To verify that the webmail server is working, we need to make sure that Apache, and MySQL are running. Enter the following command,

```
ps -ef
```

This will generate a list of the processes currently running on the system. The following lines are the ones we want to see.

```
nobody   21174   991  0 Nov09 ?        00:00:00 /usr/local/apache/bin/httpd -DSSL
nobody   21175   991  0 Nov09 ?        00:00:00 /usr/local/apache/bin/httpd -DSSL
nobody   21176   991  0 Nov09 ?        00:00:00 /usr/local/apache/bin/httpd -DSSL
nobody   21177   991  0 Nov09 ?        00:00:00 /usr/local/apache/bin/httpd -DSSL
nobody   21178   991  0 Nov09 ?        00:00:00 /usr/local/apache/bin/httpd -DSSL
nobody   21179   991  0 Nov09 ?        00:00:00 /usr/local/apache/bin/httpd -DSSL
nobody   21180   991  0 Nov09 ?        00:00:00 /usr/local/apache/bin/httpd -DSSL
nobody   21181   991  0 Nov09 ?        00:00:00 /usr/local/apache/bin/httpd -DSSL
mysql    11343 11316  0 Nov09 ?        00:00:00 /usr/libexec/mysqld --defaults-
file=/etc/my.cnf --basedir=/usr --datadir=/var/
mysql    11345 11343  0 Nov09 ?        00:00:00 /usr/libexec/mysqld --defaults-
file=/etc/my.cnf --basedir=/usr --datadir=/var/
mysql    11346 11345  0 Nov09 ?        00:00:00 /usr/libexec/mysqld --defaults-
file=/etc/my.cnf --basedir=/usr --datadir=/var/
mysql    11347 11345  0 Nov09 ?        00:00:00 /usr/libexec/mysqld --defaults-
file=/etc/my.cnf --basedir=/usr --datadir=/var/
```

These lines indicate that both Apache and MySQL are running on the system. Once you have done this we will see if the web site is accessible via the Internet. On a machine with a web browser go to https://192.168.0.4, you should be taken to a Horde webmail login screen. There will be a padlock icon at the bottom of the browser window click on that and you will be able to view the SSL certificate information. In Netscape, the security tab of the certificate window lists "Connection Encrypted: High Grade Encryption (RC4 128 bit)". In Internet Explorer, move the mouse arrow over the lock icon, and it says "SSL Secured (128 bit)". Enter your username and password, then click login. The mail window opens, and displays nothing. This is correct, as the test account has not received any mail. Click compose, and send a test message to yourself. Once you have done that, click the Address Book icon, and add an entry consisting of only a name "test" and email address "testing". Click save and then Address book icon. This takes you back to the Search page. Select Name and then enter

the word test in the search box. Click Search, and the test address book entry is displayed. Click the mail Icon, and the test email message is displayed. The Webmail server is functioning properly. The output of the SSL information in the web browsers indicates that the SSL server is working and configured properly for strong encryption. The Address Book entry verified that the Horde Framework can update the MySQL database, and display information stored there.

6.2 SSH:

In this section we will verify that SSH is running, accepting external logins, and denying root logins. As before enter,

```
ps -ef
```

the output line that we are looking for is:

```
root      2597     1  0 Nov08 ?        00:00:00 /usr/sbin/sshd
```

This line indicates that SSH is running. We will now confirm that it is accepting connections. Using an SSH client program, such as Putty, on a Windows machine. ssh to 192.168.0.4. The output will be similar to the following.

```
login as: jphilput
        --WARNING-- Unauthorized uses are prohibited, and will be
        prosecuted. All connections are logged and traced.
jphilput@192.168.0.4's password:
Last login: Mon Nov 10 02:56:31 2003 from 192.168.0.3
        --WARNING-- Unauthorized uses are prohibited, and will be
        prosecuted. All connections are logged and traced.
[jphilput@localhost jphilput]$
```

Log in again using the same method this time log in as root. The output will be similar to the following:

```
login as: root
        --WARNING-- Unauthorized uses are prohibited, and will be
        prosecuted. All connections are logged and traced.
root@192.168.0.4's password:
Access denied
root@192.168.0.4's password:
Access denied
```

Now we will verify that SSH is running in version 2 only mode. Open Putty, and enter the SSH Host Name and Port; then select the SSH category from the list on

the left side of the screen. Under "Preferred SSH protocol version" select "1 only" connect to the server, and you will see this error:

Fatal Error: SSH protocol version 1 required by user but not provided by server

The SSH program is working properly.

6.3 Sudo:

We need to make certain that the sudo command is functional and operating as it should. To do this we will need to ssh into the server as one of the users in the sudoers file, and attempt to run one of the sudo commands.

```
login as: sysadmin1
        --WARNING-- Unauthorized uses are prohibited, and will be
        prosecuted. All connections are logged and traced.
sysadmin1@192.168.0.4's password:
        --WARNING-- Unauthorized uses are prohibited, and will be
        prosecuted. All connections are logged and traced.
[sysadmin1@localhost sysadmin1]$ sudo su -
Password:
[root@localhost root]#
```

Sudo is functioning properly.

6.4 Nessus:

As mentioned above, Nessus is a testing program that uses exploit modules to test a server for vulnerabilities. We will run Nessus against the webmail server with all modules except for those which can damage the system. The Nessus server should be installed on a different machine. Log onto that machine and start Nessus with the following command.

```
nessus
```

A new window will open. Enter your Nessus username and password, and then click the "Log In" button.

Select "Plugins" from the list of tabs across the top of the screen.
Click the "Enable All" button.

Select the "Scan Options" tab.
Uncheck "Optimize the test" this option tells Nessus to try all of its modules against the machine. This makes for a longer test, but it may catch a vulnerability that would have been missed otherwise.

Uncheck "Safe Checks" - This option allows Nessus to use modules that may disable certain server services, and require a reboot.

Select the "Target Tab"
Enter 192.168.0.4 on the "Target(s):" line.

Click the button labeled,
"Start the Scan"

Nessus will begin by port scanning the server. Once this is done, it will begin the attack. When it is finished it will generate a report detailing the problems that it found.

Nessus found several things that it considers to be problems. Some of them are actual problems, while others are not. First, Nessus found several Vulnerabilities in SSH and SSL. All of these vulnerabilities have been patched in the most recent updates to the software. Nessus only used banners to verify the version of Openssh running on the server. The updates that were downloaded from RedHat, and installed earlier in the build, patched these programs to their most recent versions. They are not vulnerable to the exploits listed by Nessus. RedHat does not update the software version number when a new patch is released. This means that the banners display an older software version, even though the server is running the most recent versions of SSH and SSL.

```
[jphilput@localhost jphilput]$ rpm -q openssh-server
openssh-server-3.1p1-14
[jphilput@localhost jphilput]$ rpm -q openssh
openssh-3.1p1-14
```

The most recent version of the software that is known to be vulnerable is 3.1p1-13.

Nessus also found several vulnerabilities in the web server. The first of these is that User Directories are turned on. The use of these can allow information gathering when a malicious user tries to connect to the directory. We will disable this feature since it is not used.

```
vi /path/to/httpd.conf
```

Comment out the UserDir lines.

```
# UserDir: The name of the directory which is appended onto a user's home
# directory if a ~user request is received.
#
#<IfModule mod_userdir.c>
#    UserDir public_html
#</IfModule>
```

:wq

/path/to/apachectl restart

Nessus found several other problems which are not actually problems. Nessus believes that the server is running an unpatched version of OpenSSL. This is the same problem with RedHat's patch naming conventions as occurred with Openssh. The version of OpenSSL running on this server is the most recent.

[root@localhost conf]# rpm -q openssl
openssl-0.9.6b-35.7

Another error was that, Nessus was able to see the servers NAT IP address. This was an expected error, since we were running Nessus against the server from the Production Network. Nessus also had problems with some of Horde's PHP files being readable from the web. While this could enable some information gathering specific only to Horde itself, these php scripts must be viewable from the web for Horde to function. The last vulnerability found by Nessus was that the server's IPChains firewall does not discard TCP SYN packets with the FIN flag set. This could lead to a potential denial of service attack. The webmail server has only a basic firewall setup. This type of incoming packet is blocked by the Production Network firewall. All of the other Items listed in the Nessus report are informational, and must be available for the server to function properly.

6.5 CIS Security Linux Benchmark:

The last verification tool that we will run on this system is CI Security's Linux Benchmarking Tool. When this program is installed and run on the system, it goes through the server settings looking for potential security problems. The tool then generates a score from 1 to 10 with 10 being the most secure, and 1 being the least secure. Before we can run the program, we will need to download and install it. The tool can be downloaded from http://www.cisecurity.com. Once you have downloaded it, make sure to run

md5sum cis-linux.tar.gz.tar

and check the output against the md5sum file on http://www.cisecurity.com. Once you have downloaded and verified the file, we will need to install it.

tar zxpf cis-linux.tar.gz.tar
cd cis
rpm -U CISscan-1.4.2-1.0.i386.rpm

Now that the software is installed, we can run it with the following command.

/usr/local/CIS/cis-scan

The tool will now begin to run It's output will look like this:

```
[root@localhost cis]# /usr/local/CIS/cis-scan

********************************************************************************
****************** CIS Security Benchmark Checker v1.4.2 ******************
*                                                                              *
* Lead Developer                                : Jay Beale        *
* Benchmark Coordinator and Gadfly                    : Hal Pomeranz      *
*                                                                              *
* Copright 2001 - 2003 The Center for Internet Security  www.cisecurity.org *
*                                                                              *
* Please send feedback to linux-scan@cisecurity.org.                           *
********************************************************************************

        Investigating system...this will take a few minutes...

                        ******

Now a final check for non-standard world-writable files, Set-UID and Set-GID
programs -- this can take a whole lot of time if you have a large filesystem.
Your score if there are no extra world-writable files or SUID/SGID programs
found will be 8.12 / 10.00 .  If there are extra SUID/SGID programs or
world-writable files, your score could be as low as 7.81 / 10.00 .

       You can hit CTRL-C at any time to stop at this remaining step.

The preliminary log can be found at: /usr/local/CIS/cis-most-recent-log

                        ******

         Rating = 8.12 / 10.00

********************************************************************************
```

This report has found several security problems that should be fixed at this point. The SSH server is already configured for use of only SSH version 2 connections, however, for greater redundancy we need to add the line "Protocol 2" to the ssh_config file.

```
        vi /etc/ssh/ssh_config
Under "Hosts *" add the line "Protocol 2"
        Host *
```

Protocol 2

We will also need to set the umask to 022 in /etc/rc.d/init.d/functions

vi /etc/rc.d/init.d/functions
Change the umask to 022
:wq

Lastly, we should set the minimum password age to 7 days.

vi /etc/login.defs
set the pass min line as follows.
PASS_MIN_DAYS 7

The rest of the settings that the benchmark tool has listed as problems or potential problems are needed for the system to function properly as a webmail server. We will run the Linux benchmark tool one more time to get the final score for this server.

```
[root@localhost cis]# /usr/local/CIS/cis-scan

*****************************************************************************
****************** CIS Security Benchmark Checker v1.4.2 ********************
*                                                                           *
* Lead Developer                          : Jay Beale                       *
* Benchmark Coordinator and Gadfly        : Hal Pomeranz                    *
*                                                                           *
* Copright 2001 - 2003 The Center for Internet Security  www.cisecurity.org *
*                                                                           *
* Please send feedback to linux-scan@cisecurity.org.                        *
*****************************************************************************

        Investigating system...this will take a few minutes...

                        ******
```

Now a final check for non-standard world-writable files, Set-UID and Set-GID programs -- this can take a whole lot of time if you have a large filesystem. Your score if there are no extra world-writable files or SUID/SGID programs found will be 8.44 / 10.00 . If there are extra SUID/SGID programs or world-writable files, your score could be as low as 8.12 / 10.00 .

You can hit CTRL-C at any time to stop at this remaining step.

The preliminary log can be found at: /usr/local/CIS/cis-most-recent-log

******

Rating = 8.44 / 10.00

This server rates an 8.44 out of a possible 10. If we continue to follow these procedures, and keep up with ongoing maintenance, we should be able to keep this machine in its secure and usable state for the foreseeable future.

7. References:

Garrels, Machtelt. "Introduction to Linux: A Hands on Guide" 16 September 2003
URL: http://www.tldp.org/LDP/intro-linux/html/sect_03_04.html#sect_03_04_02_02

Network World Fusion "Networking Encyclopedia" 2 September 2002
URL: http://www.nwfusion.com/links/Encyclopedia/S/674.html

Hines, Eric. "Complete Reference Guide to Creating a Remote Log Server" 22 August 2000.
URL: http://www.linuxsecurity.com/feature_stories/remote_logserver-3.html

PHP.net "PHP Manual: Using Register Globals" 21 August 2003
URL: http://www.php.net/manual/en/security.registerglobals.php

Engelschall, Ralf S. "mod_ssl 2.8, User Manual The Apache Interface to OpenSSL" 2001
URL: http://www.modssl.org/docs/2.8/ssl_howto.html#ToC2

Rostetter, Eric. "Horde Administrator's FAQ" September 2003
URL: http://www.horde.org/faq/admin/print.php

mod_ssl INSTALL document

RedHat Linux man pages

8. Appendix A - Apache Make Certificate Detail

```
 make[1]: Entering directory
`/usr/local/src/apache_1.3.28/src'
SSL Certificate Generation Utility (mkcert.sh)
Copyright (c) 1998-2000 Ralf S. Engelschall, All Rights
Reserved.

Generating test certificate signed by Snake Oil CA [TEST]
WARNING: Do not use this for real-life/production systems
_____________________________________________________________
___________

STEP 0: Decide the signature algorithm used for certificate
The generated X.509 CA certificate can contain either
RSA or DSA based ingredients. Select the one you want to
use.
Signature Algorithm ((R)SA or (D)SA) [R]:
_____________________________________________________________
___________

STEP 1: Generating RSA private key (1024 bit) [server.key]
168271 semi-random bytes loaded
Generating RSA private key, 1024 bit long modulus
.......++++++
....++++++
e is 65537 (0x10001)
_____________________________________________________________
___________

STEP 2: Generating X.509 certificate signing request
[server.csr]
Using configuration from .mkcert.cfg
You are about to be asked to enter information that will be
incorporated
into your certificate request.
What you are about to enter is what is called a
Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
1. Country Name             (2 letter code) [XY]:
2. State or Province Name   (full name)     [Snake Desert]:
3. Locality Name            (eg, city)      [Snake Town]:
4. Organization Name        (eg, company)   [Snake Oil,
Ltd]:
```

```
5. Organizational Unit Name (eg, section)   [Webserver
Team]:
6. Common Name               (eg, FQDN)
[www.snakeoil.dom]:
7. Email Address             (eg, name@FQDN)
[www@snakeoil.dom]:
8. Certificate Validity      (days)          [365]:
```

```
STEP 3: Generating X.509 certificate signed by Snake Oil CA
[server.crt]
Certificate Version (1 or 3) [3]:
Signature ok
subject=/C=XY/ST=Snake Desert/L=Snake Town/O=Snake Oil,
Ltd/OU=Webserver
Team/CN=www.snakeoil.dom/Email=www@snakeoil.dom
Getting CA Private Key
Verify: matching certificate & key modulus
read RSA key
Verify: matching certificate signature
../conf/ssl.crt/server.crt: /C=XY/ST=Snake Desert/L=Snake
Town/O=Snake Oil, Ltd/OU=Certificate Authority/CN=Snake Oil
CA/Email=ca@snakeoil.dom
error 10 at 1 depth lookup:Certificate has expired
OK
```

```
STEP 4: Enrypting RSA private key with a pass phrase for
security [server.key]
The contents of the server.key file (the generated private
key) has to be
kept secret. So we strongly recommend you to encrypt the
server.key file
with a Triple-DES cipher and a Pass Phrase.
Encrypt the private key now? [Y/n]:
read RSA key
writing RSA key
Enter PEM pass phrase:
Verifying password - Enter PEM pass phrase:
Fine, you're using an encrypted RSA private key.
```

RESULT: Server Certification Files

```
o  conf/ssl.key/server.key
   The PEM-encoded RSA private key file which you configure
   with the 'SSLCertificateKeyFile' directive
(automatically done
   when you install via APACI). KEEP THIS FILE PRIVATE!

o  conf/ssl.crt/server.crt
   The PEM-encoded X.509 certificate file which you
configure
   with the 'SSLCertificateFile' directive (automatically
done
   when you install via APACI).


o  conf/ssl.csr/server.csr
   The PEM-encoded X.509 certificate signing request file
which
   you can send to an official Certificate Authority (CA)
in order
   to request a real server certificate (signed by this CA
instead
   of our demonstration-only Snake Oil CA) which later can
replace
   the conf/ssl.crt/server.crt file.

WARNING: Do not use this for real-life/production systems

make[1]: Leaving directory
`/usr/local/src/apache_1.3.28/src'
```

9. Appendix B - Nessus Scan Report

```
Nessus Scan Report
------------------



SUMMARY

 - Number of hosts which were alive during the test : 1
 - Number of security holes found : 4
 - Number of security warnings found : 7
 - Number of security notes found : 11



TESTED HOSTS

 192.168.0.4 (Security holes found)



DETAILS

+ 192.168.0.4 :
 . List of open ports :
   o ssh (22/tcp) (Security hole found)
   o https (443/tcp) (Security hole found)
   o general/tcp (Security warnings found)
   o general/udp (Security notes found)
   o general/icmp (Security warnings found)

 . Vulnerability found on port ssh (22/tcp) :



    You are running a version of OpenSSH older than OpenSSH
3.2.1

    A buffer overflow exists in the daemon if AFS is
enabled on
    your system, or if the options KerberosTgtPassing or
    AFSTokenPassing are enabled.  Even in this scenario,
the
    vulnerability may be avoided by enabling
UsePrivilegeSeparation.
```

```
    Versions prior to 2.9.9 are vulnerable to a remote root
    exploit. Versions prior to 3.2.1 are vulnerable to a
local
    root exploit.

    Solution :
    Upgrade to the latest version of OpenSSH

    Risk factor : High
    CVE : CVE-2002-0575
    BID : 4560

 . Vulnerability found on port ssh (22/tcp) :



    You are running a version of OpenSSH which is older
than 3.4

    There is a flaw in this version that can be exploited
remotely to
    give an attacker a shell on this host.

    Note that several distribution patched this hole
without changing
    the version number of OpenSSH. Since Nessus solely
relied on the
    banner of the remote SSH server to perform this check,
this might
    be a false positive.

    If you are running a RedHat host, make sure that the
command :
              rpm -q openssh-server

    Returns :
     openssh-server-3.1p1-6


    Solution : Upgrade to OpenSSH 3.4 or contact your
vendor for a patch
    Risk factor : High
    CVE : CVE-2002-0639, CVE-2002-0640
    BID : 5093

 . Vulnerability found on port ssh (22/tcp) :
```

```
    You are running a version of OpenSSH which is older
than 3.7.1

    Versions older than 3.7.1 are vulnerable to a flaw in
the buffer management
    functions which might allow an attacker to execute
arbitrary commands on
     this
    host.

    An exploit for this issue is rumored to exist.


    Note that several distribution patched this hole
without changing
    the version number of OpenSSH. Since Nessus solely
relied on the
    banner of the remote SSH server to perform this check,
this might
    be a false positive.

    If you are running a RedHat host, make sure that the
command :
                rpm  -q openssh-server

    Returns :
     openssh-server-3.1p1-13 (RedHat 7.x)
     openssh-server-3.4p1-7  (RedHat 8.0)
     openssh-server-3.5p1-11 (RedHat 9)

    So lution : Upgrade to OpenSSH 3.7.1
    See also :
     http://marc.theaimsgroup.com/?l=openbsd-
misc&m=106375452423794&w=2
        http://marc.theaimsgroup.com/?l=openbsd-
misc&m=106375456923804&w=2
    Risk factor : High
    CVE : CAN-2003-0693, CAN-2003-0695
    BID : 8628

 . Warning found on port ssh (22/tcp)


    You are running OpenSSH-portable 3.6.1p1 or older.
```

```
    If PAM support is enabled, an attacker may use a flaw
in this version
    to determine the existence or a given login name by
comparing the times
    the remote sshd daemon takes to refuse a bad password
for a non-existant
    login compared to the time it takes to refuse a bad
password for a
    valid login.

    An attacker may use this flaw to set up  a brute force
attack against
    the remote host.

    *** Nessus did not check whether the remote SSH daemon
is actually
    *** using PAM or not, so this might be a false positive

    Solution : Upgrade to OpenSSH-portable 3.6.1p2 or newer
    Risk Factor : Low
    CVE : CAN-2003-0190
    BID : 7482, 7467, 7342

 . Warning found on port ssh (22/tcp)



    You are running OpenSSH-portable 3.6.1 or older.

    There is a flaw in this version which may allow an
attacker to
    bypass the access controls set by the administrator of
this server.

    OpenSSH features a mechanism which can restrict the
list of
    hosts a given user can log from by specifying a pattern
    in the user key file (ie: *.mynetwork.com would let a
user
    connect only from the local network).

    However there is a flaw in the way OpenSSH does reverse
DNS lookups.
    If an attacker configures his DNS server to send a
numeric IP address
```

```
    when a reverse lookup is performed, he may be able to
circumvent
    this mechanism.

    Solution : Upgrade to OpenSSH 3.6.2 when it comes out
    Risk Factor : Low
    CVE : CAN-2003-0386
    BID : 7831

 . Information found on port ssh (22/tcp)


    An ssh server is running on this port

 . Information found on port ssh (22/tcp)


    Remote SSH version : SSH-2.0-OpenSSH_3.1p1

 . Information found on port ssh (22/tcp)


    The remote SSH daemon supports the following versions
of the
    SSH protocol :

      . 1.99
      . 2.0


  . Vulnerability found on port https (443/tcp) :


    The following requests seem to allow the reading of
    sensitive files or XSS. You should manually try them to
see if anything bad
     happens :
    /horde/css.php?app=<script>alert('foo');</script>

/horde/imp/redirect.php?imapuser=nessus&button=[Log&in]=nes
sus&maildomain=192.168.0.10&pass=nessus&namespace=nessus&ur
l=nessus&port=143&realm=nessus&actionID=105&mailbox=INBOX&f
olders=nessus&server=192.168.0.10&new_lang=nessus&protocol=
imap/notls&Horde=<script>alert('foo');</script>
/horde/menu.php?Horde=<script>alert('foo');</script>
/horde/login.php?Horde=<script>alert('foo');</script>
```

```
  . Warning found on port https (443/tcp)


    This web server leaks a private IP address through its
HTTP headers :
     192.168.0.4

    This may expose internal IP addresses that are usually
hidden or masked
    behind a Network Address Translation (NAT) Firewall or
proxy server.

    There is a known issue with IIS 4.0 doing this in its
default configuration.
      See
http://support.microsoft.com/support/kb/articles/Q218/1/80.
ASP

    See the Bugtraq reference for a full discussion.

    Risk factor : Low
    CVE : CAN-2000-0649
    BID : 1499

 . Warning found on port https (443/tcp)


    The remote host is using a version of OpenSSL which is
    older than 0.9.6j or 0.9.7b

    This version is vulnerable to a timing based attack
which may
    allow an attacker to guess the content of fixed data
blocks and
    may eventually be able to guess the value of the
private RSA key
    of the server.

    An attacker may use this implementation flaw to sniff
the
    data going to this host and decrypt some parts of it,
as well
```

```
    as impersonate your server and perform man in the
middle attacks.

    *** Nessus solely relied on the banner of the remote
host
    *** to issue this warning

    See also :
http://www.openssl.org/news/secadv_20030219.txt
        http://lasecwww.epfl.ch/memo_ssl.shtml
        http://eprint.iacr.org/2003/052/

    Solution : Upgrade to version 0.9.6j (0.9.7b) or newer
    Risk factor : Medium
    CVE : CAN-2003-0078, CAN-2003-0131
    BID : 6884, 7148

 . Warning found on port https (443/tcp)


    The SSLv2 server offers 5 strong ciphers, but also
    0 medium strength and 2 weak "export class" ciphers.
    The weak/medium ciphers may be chosen by an export-
grade
    or badly configured client software. They only offer a
    limited protection against a brute force attack

    Solution: disable those ciphers and upgrade your client
    software if necessary

 . Information found on port https (443/tcp)


    A SSLv2 server answered on this port


 . Information found on port https (443/tcp)


    A web server is running on this port through SSL

 . Information found on port https (443/tcp)


    The following directories were discovered:
    /cgi-bin, /config, /docs, /icons, /lib, /login,
/manual, /scripts, /status,
```

```
     /templates, /test

 . Information found on port https (443/tcp)


    The following CGI have been discovered :

    Syntax : cginame (arguments [default value])

    /horde/login.php (Horde
[259fe0dacec10c4d9c800863ec95f482] )
    /horde/menu.php (Horde
[259fe0dacec10c4d9c800863ec95f482] )
    /horde/imp/redirect.php (imapuser [] Horde
     [2ecd3ea42f664c5cdc02c8224998df2d] button [Log in]
maildomain
     [192.168.0.10] pass [] namespace [] url [] port [143]
realm [] actionID
     [105] mailbox [INBOX] folders [] server [192.168.0.10]
new_lang [] protocol
     [imap/notls] )
    /horde/imp/graphics/ (D [A] M [A] N [A] D=D [] S [A] )
    /horde/css.php (app [imp] )


    Directory index found at /horde/imp/graphics/


 . Information found on port https (443/tcp)


    The remote web server type is :

    Apache/1.3.29 (Unix) PHP/4.3.3 mod_ssl/2.8.16
OpenSSL/0.9.6b

    Solution : You can set the directive 'ServerTokens
Prod' to limit
    the information emanating from the server in its
response headers.

 . Information found on port https (443/tcp)


    An information leak occurs on Apache based web servers
    whenever the UserDir module is enabled. The
vulnerability allows an external
```

```
    attacker to enumerate existing accounts by requesting
access to their home
    directory and monitoring the response.


    Solution:
    1) Disable this feature by changing 'UserDir
public_html' (or whatever) to
    'UserDir  disabled'.

    Or

    2) Use a RedirectMatch rewrite rule under Apache --
this works even if there

    is no such  entry in the password file, e.g.:
    RedirectMatch ^/~(.*)$ http://my-target-
webserver.somewhere.org/$1

    Or

    3) Add into httpd.conf:
    ErrorDocument 404 http://localhost/sample.html
    ErrorDocument 403 http://localhost/sample.html
    (NOTE: You need to use a FQDN inside the URL for it to
work properly).

    Additional Information:
    http://www.securiteam.com/unixfocus/5WP0C1F5FI.html


    Risk factor : Low
    CVE : CAN-2001-1013
    BID : 3335

 . Information found on port https (443/tcp)


    Here is the SSLv2 server certificate:
    Certificate:
        Data:
            Version: 3 (0x2)
            Serial Number: 1 (0x1)
            Signature Algorithm: md5WithRSAEncryption
            Issuer: C=XY, ST=Snake Desert, L=Snake Town,
O=Snake Oil, Ltd,
```

```
     OU=Certificate Authority, CN=Snake Oil
CA/Email=ca@snakeoil.dom
            Validity
                Not Before: Nov  2 23:08:42 2003 GMT
                Not After : Nov  1 23:08:42 2004 GMT
            Subject: C=XY, ST=Snake Desert, L=Snake Town,
O=Snake Oil, Ltd,
     OU=Webserver Team,
CN=www.snakeoil.dom/Email=www@snakeoil.dom
            Subject Public Key Info:
                Public Key Algorithm: rsaEncryption
                RSA Public Key: (1024 bit)
                    Modulus (1024 bit):

00:d5:bd:5a:6e:6b:55:df:c6:56:3f:a3:94:34:13:

55:ad:9a:bb:2c:1f:9e:f6:73:f7:82:41:cd:e2:26:

c1:02:6c:04:50:72:9e:3a:1d:0a:ed:c4:30:30:e3:

4a:05:c0:53:0c:45:c8:08:9c:78:89:7c:bd:2d:c3:

34:b2:1e:38:84:0d:66:b8:62:15:03:37:ee:be:a8:

76:b3:44:71:b1:e3:f7:ca:b8:18:95:4f:72:92:2e:

6e:a0:68:4a:04:27:c6:71:b8:31:ea:96:13:8a:7a:

79:5f:68:6c:75:07:25:bf:af:54:70:5f:9e:24:66:
                    e7:dc:ca:51:6f:6a:59:35:89
                Exponent: 65537 (0x10001)
            X509v3 extensions:
                X509v3 Subject Alternative Name:
                    email:www@snakeoil.dom
                Netscape Comment:
                    mod_ssl generated test server
certificate
                Netscape Cert Type:
                    SSL Server
        Signature Algorithm: md5WithRSAEncryption

1d:1b:57:24:73:e4:7b:e9:d6:9b:fc:d1:7e:51:90:de:b4:ed:

d5:69:43:fd:17:cb:c7:31:77:94:1f:a0:01:6d:85:99:52:b2:

b5:ae:d5:02:0c:bd:2b:f0:1d:8c:ab:e5:0e:4b:57:38:99:99:
```

```
97:33:55:6f:9a:e6:41:1f:1b:08:a0:c0:d2:57:84:38:bc:e1:

0c:06:1c:6b:a6:d8:3c:3c:ed:81:cf:e3:67:5e:04:50:0a:bf:

71:38:9a:07:a6:0c:47:2a:d2:fa:e6:ca:7d:8f:18:82:74:8d:

c5:ad:19:7c:74:cc:f2:5b:dd:54:06:d4:12:70:37:7d:91:2e:
           e6:70
```

. Information found on port https (443/tcp)

```
   Here is the list of available SSLv2 ciphers:
   RC4-MD5
   EXP-RC4-MD5
   RC2-CBC-MD5
   EXP-RC2-CBC-MD5
   DES-CBC-MD5
   DES-CBC3-MD5
   RC4-64-MD5
```

. Information found on port https (443/tcp)

```
   This SSLv2 server does not accept SSLv3 connections.
   This SSLv2 server does not accept TLSv1 connections.
```

. Warning found on port general/tcp

```
   The remote host does not discard TCP SYN packets which
   have the FIN flag set.

   Depending on the kind of firewall you are using, an
   attacker may use this flaw to bypass its rules.

   See also :
http://archives.neohapsis.com/archives/bugtraq/2002-
10/0266.html
```

http://www.kb.cert.org/vuls/id/464113

Solution : Contact your vendor for a patch
Risk factor : Medium
BID : 7487

. Information found on port general/tcp

Remote OS guess : Linux Kernel 2.4.0 - 2.5.20

CVE : CAN-1999-0454

. Information found on port general/udp

For your information, here is the traceroute to 192.168.0.4 :
192.168.0.4

. Warning found on port general/icmp

The remote host answers to an ICMP timestamp request. This allows an attacker to know the date which is set on your machine.

This may help him to defeat all your time based authentication protocols.

Solution : filter out the ICMP timestamp requests (13), and the outgoing ICMP timestamp replies (14).

Risk factor : Low
CVE : CAN-1999-0524

This file was generated by the Nessus Security Scanner

10. Appendix C - CIS Security Benchmark Log

```
*** CIS Ruler Run ***
Starting at time 20031110-12:47:38

Positive: 1.1 System appears to have been patched within
the last month.
Positive: 1.2 SSH client and server are configured well.
Positive: 2.1 inetd/xinetd is not listening on any of the
miscellaneous ports ch
ecked in this item.
Positive: 2.2 telnet is deactivated.
Positive: 2.3 ftp is deactivated.
Positive: 2.4 rsh, rcp and rlogin are deactivated.
Positive: 2.5 tftp is deactivated.
Positive: 2.6 imap is deactivated.
Positive: 2.7 POP server is deactivated.
Positive: 3.1 Found a good daemon umask of 022 in
/etc/rc.d/init.d/functions.
Positive: 3.2 inetd has been deactivated.
Positive: 3.3 Mail daemon is not listening on TCP 25.
Positive: 3.4 Graphical login is deactivated.
Positive: 3.5 X Font Server (xfs) script has been
deactivated
Positive: 3.6 Miscellaneous scripts are all turned off.
Positive: 3.7 Windows compatibility servers (samba) have
been deactivated.
Positive: 3.8 NFS Server script nfs is deactivated.
Positive: 3.9 This machine isn't being used as an NFS
client.
Positive: 3.10 NIS Client processes are deactivated.
Positive: 3.11 NIS Server processes are deactivated.
Positive: 3.12 RPC rc-script has been deactivated.
Positive: 3.13 netfs rc script is deactivated.
Positive: 3.14 printing daemon is deactivated.
Negative: 3.15 Web server not deactivated.
Positive: 3.16 SNMP daemon is deactivated.
Positive: 3.17 DNS server is deactivated.
Positive: 3.18 SQL database server is deactivated.
Positive: 3.19 Webmin GUI-based system administration
daemon deactivated.
Positive: 3.20 Squid web cache daemon deactivated.
Positive: 3.21 Kudzu hardware detection program has been
deactivated.
```

```
Negative: 4.1
/proc/sys/net/ipv4/conf/eth0/accept_source_route should be
set to
0.
Negative: 4.1
/proc/sys/net/ipv4/conf/lo/accept_source_route should be
set to 0.
Negative: 4.1 /proc/sys/net/ipv4/conf/eth0/accept_redirects
should be set to 0.
Negative: 4.1 /proc/sys/net/ipv4/conf/lo/accept_redirects
should be set to 0.
Negative: 4.1 /proc/sys/net/ipv4/conf/eth0/secure_redirects
should be set to 0.
Negative: 4.1 /proc/sys/net/ipv4/conf/lo/secure_redirects
should be set to 0.
Negative: 4.2 /proc/sys/net/ipv4/conf/eth0/send_redirects
should be set to 0.
Negative: 4.2 /proc/sys/net/ipv4/conf/lo/send_redirects
should be set to 0.
Positive: 5.1 syslog captures authpriv messages.
Positive: 5.2 FTP server is configured to do full logging.
Positive: 5.3 All logfile permissions and owners match
benchmark recommendations
.
Negative: 6.1 /boot is not mounted nodev.
Positive: 6.2 /etc/fstab mounts all removable filesystems
nosuid and nodev.
Positive: 6.3 Users cannot mount removable media.
Positive: 6.4 password and group files have right
permissions and owners.
Positive: 6.5 all temporary directories have sticky bits
set.
Positive: 7.1 rhosts authentication totally deactivated in
PAM.
Positive: 7.2 /etc/hosts.equiv and root's .rhosts/.shosts
files either don't exi
st, are zero size or are links to /dev/null.
Positive: 7.3 FTP daemons do not permit system users to use
FTP.
Positive: 7.4 X11 Server is blocked from listening on TCP
port 6000.
Negative: 7.5 Couldn't open cron.allow
Negative: 7.5 Couldn't open at.allow
Negative: 7.6 The permissions on /etc/crontab are not
sufficiently restrictive.
Positive: 7.7 All authorized-use-only warning banners are
in place.
```

```
Negative: 7.9 /etc/securetty has a non console or tty 1-6
line: vc/7.
Negative: 7.9 /etc/securetty has a non console or tty 1-6
line: vc/8.
Negative: 7.9 /etc/securetty has a non console or tty 1-6
line: vc/9.
Negative: 7.9 /etc/securetty has a non console or tty 1-6
line: vc/10.
Negative: 7.9 /etc/securetty has a non console or tty 1-6
line: vc/11.
Negative: 7.9 /etc/securetty has a non console or tty 1-6
line: tty7.
Negative: 7.9 /etc/securetty has a non console or tty 1-6
line: tty8.
Negative: 7.9 /etc/securetty has a non console or tty 1-6
line: tty9.
Negative: 7.9 /etc/securetty has a non console or tty 1-6
line: tty10.
Negative: 7.9 /etc/securetty has a non console or tty 1-6
line: tty11.
Positive: 7.10 GRUB is password-protected.
Positive: 7.10 GRUB is password-protected.
Positive: 7.11 Single user mode requires a root password.
Positive: 7.12 /etc/exports is empty or doesn't exist, so
it doesn't need to be
tuned for privports.
Negative: 8.1 bin has a valid shell of /sbin/nologin.
Remember, the /sbin/nolog
in shell, when found in /etc/shells, leaves a user
potentially able to use FTP.
Negative: 8.1 daemon has a valid shell of /sbin/nologin.
Remember, the /sbin/no
login shell, when found in /etc/shells, leaves a user
potentially able to use FT
P.
Negative: 8.1 nobody has a valid shell of /sbin/nologin.
Remember, the /sbin/no
login shell, when found in /etc/shells, leaves a user
potentially able to use FT
P.
Negative: 8.1 vcsa has a valid shell of /sbin/nologin.
Remember, the /sbin/nolo
gin shell, when found in /etc/shells, leaves a user
potentially able to use FTP.
Negative: 8.1 rpm has a valid shell of /bin/bash.
Negative: 8.1 mysql has a valid shell of /bin/bash.
Positive: 8.2 All users have passwords
```

```
Negative: 8.3 User jphilput should have a minimum password
life of at least 7 da
ys.
Negative: 8.3 User jphilput should have a maximum password
life of between 1 and
 90 days.
Negative: 8.3 User otheruser should have a minimum password
life of at least 7 d
ays.
Negative: 8.3 User otheruser should have a maximum password
life of between 1 an
d 90 days.
Negative: 8.3 User otheruser1 should have a minimum
password life of at least 7
days.
Negative: 8.3 User otheruser2 should have a minimum
password life of at least 7
days.
Negative: 8.3 User otheruser3 should have a minimum
password life of at least 7
days.
Negative: 8.3 User sysadmin1 should have a minimum password
life of at least 7 d
ays.
Positive: 8.4 There were no +: entries in passwd, shadow or
group maps.
Positive: 8.5 Only one UID 0 account AND it is named root.
Positive: 8.6 root's PATH is clean of group/world writable
directories or the cu
rrent-directory link.
Positive: 8.7 No user's home directory is world or group
writable.
Positive: 8.8 No group or world-writable dotfiles in user
home directories!
Positive: 8.9 No user has a .netrc file.
Negative: 8.10 Current umask setting in file /etc/profile
is 000 -- it should be
 stronger to block world-read/write/execute.
Negative: 8.10 Current umask setting in file /etc/profile
is 000 -- it should be
 stronger to block group-read/write/execute.
Negative: 8.10 Current umask setting in file /etc/csh.login
is 000 -- it should
be stronger to block world-read/write/execute.
Negative: 8.10 Current umask setting in file /etc/csh.login
is 000 -- it should
be stronger to block group-read/write/execute.
```

```
Negative: 8.10 Current umask setting in file /etc/bashrc is
022 -- it should be
stronger to block world-read/write/execute.
Negative: 8.10 Current umask setting in file /etc/bashrc is
022 -- it should be
stronger to block group-read/write/execute.
Negative: 8.10 Current umask setting in file /etc/csh.cshrc
is 002 -- it should
be stronger to block world-read/write/execute.
Negative: 8.10 Current umask setting in file /etc/csh.cshrc
is 002 -- it should
be stronger to block group-read/write/execute.
Negative: 8.10 Current umask setting in file
/root/.bash_profile is 000 -- it sh
ould be stronger to block world-read/write/execute.
Negative: 8.10 Current umask setting in file
/root/.bash_profile is 000 -- it sh
ould be stronger to block group-read/write/execute.
Negative: 8.10 Current umask setting in file /root/.bashrc
is 000 -- it should b
e stronger to block world-read/write/execute.
Negative: 8.10 Current umask setting in file /root/.bashrc
is 000 -- it should b
e stronger to block group-read/write/execute.
Negative: 8.10 Current umask setting in file /root/.cshrc
is 000 -- it should be
 stronger to block world-read/write/execute.
Negative: 8.10 Current umask setting in file /root/.cshrc
is 000 -- it should be
 stronger to block group-read/write/execute.
Negative: 8.10 Current umask setting in file /root/.tcshrc
is 000 -- it should b
e stronger to block world-read/write/execute.
Negative: 8.10 Current umask setting in file /root/.tcshrc
is 000 -- it should b
e stronger to block group-read/write/execute.
Positive: 8.11 Coredumps are deactivated.
Preliminary rating given at time: Mon Nov 10 12:47:39 2003

        Preliminary rating = 8.12 / 10.00

Positive: 6.6 No non-standard world-writable files.
Positive: 6.7 No non-standard SUID/SGID programs found.
Ending run at time: Mon Nov 10 12:47:40 2003

        Final rating = 8.44 / 10.00
```

Last Updated: May 23rd, 2017

Upcoming Training

SANS London July 2017	**London, United Kingdom**	**Jul 03, 2017 - Jul 08, 2017**	**Live Event**
SANSFIRE 2017	**Washington, DC**	**Jul 22, 2017 - Jul 29, 2017**	**Live Event**
SANS Network Security 2017	**Las Vegas, NV**	**Sep 10, 2017 - Sep 17, 2017**	**Live Event**
SANS OnDemand	**Online**	**Anytime**	**Self Paced**
SANS SelfStudy	**Books & MP3s Only**	**Anytime**	**Self Paced**