Global Information Assurance Certification Paper

Installing and Securing a Shell Access Server Using Red Hat 6.2 Linux

Steve Gibson

The following checklist will serve as a guide for installing and securing a Red Hat 6.2 Linux system intended to be used for remote shell access. Users will be able to connect remotely using only the ssh suite of applications (i.e. there will be no access to the server via ftp, http, smtp, pop3, or imap).

Access to standard services (mail, news, www) will be available to users once they login to the server via ssh. Incoming mail can be retrieved via remote POP3 or IMAP mail servers. Outbound mail will be handled by sendmail (running in non-daemon mode).

Users will be able to make use of scripting languages such as perl, awk, and shell scripts, but will not have access to compilers. Printing will not be supported on this server.

The checklist is broken down into five major sections:

1. Installation of the Red Hat 6.2 Linux Operating System
2. Updating the base installation
3. Install additional software
4. Securing and optimizing the system
5. Initial backup

Examples provided in this checklist are based on installing to an Intel-based system with the following components: Pentium II 350mhz, 128M RAM, 10G IDE hard drive, 3com 905b 10/100 NIC, CDROM and floppy drive. Output and certain settings may vary somewhat from those shown depending on the specifications of the hardware you use.

Step 1:
Installation of the Red Hat 6.2 Linux Operating System

Before starting, make sure that the computer you will be installing on is not actually connected to the network. Simply unplugging the Ethernet cable from the NIC will suffice. You will need to connect to the network later in the process, so be sure that your system is located in an area where you will be able to plug in to the network.

1.1) ___ Verify that the computer is not physically connected to the network.

1.2) ___ Boot from the Red Hat 6.2 installation CD.

If your BIOS will not allow you to boot from the CDROM drive, use the Red Hat 6.2 Boot Floppy that comes with the distribution. Hit 'enter' at the LILO boot prompt.

1.3) Beginning install

a) ___ Select appropriate language, keyboard, and mouse settings suitable for your hardware.

1.4) Install Options

a) ___ Select 'custom'

1.5) ___ Partition layout:

For a shell account server, using a single 10GB hard disk, a reasonable partition table might look like this:

/	512M
/usr	1024M
/home	5263M (large enough for several user accounts)
/usr/local	1024M (to keep add-on applcations separate from the core binaries)
/var	1024M (if sending logs to a remote syslog, reduce this considerably)
/tmp	512M
<swap>	256M (swap generally is set at RAM * 2)

Be sure to designate a separate /tmp and /var partition to help deter denial of service attacks based on filling up the /tmp and /var volumes. This is particularly true in the case of shell account servers where users have direct access to the /tmp directory.

NOTE: If this server will be used to service a large number of user accounts, a separate hard disk should be considered for housing the /home partition.

Fill in your partition layout below:

Partition	**Size**
/	_____
/usr	_____
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____

1.6) Partitions to format

a) ___ Checkmark all partitions for formatting
b) ___ Do not check for bad blocks

Modern IDE drives map bad blocks internally. Checking for bad blocks takes a LONG time. Unless you are using a very OLD drive and it would make you feel better to run the check, don't bother.

1.7) LILO configuration

Stick with the defaults.

a) ___ Create boot disk (yes)
b) ___ Install LILO on Master Boot Record (yes)

1.8) Network configuration

a) ___ Deselect DHCP configuration

The server should use a static IP address if possible.

b) ___ Activate on boot? (select yes)

_____ IP address
_____ Netmask
_____ Network
_____ Broadcast
_____ Hostname
_____ Gateway
_____ DNS primary
_____ DNS secondary
_____ DNS tertiary

1.9) Time zone selection

If your hardware clock is set to Universal Time Coordinated (UTC), which is Greenwich Mean Time (GMT) updated with leap seconds, be sure to check the box labeled "System clock uses UTC". Then select the proper time zone for the location where the box will be operating or UTC if all your systems are on GMT.

a) Time Zone:_____

1.10) Account Configuration

a) ___ Root password

Select a strong root password, be sure that it is not a word found in the dictionary (of any language). Good passwords use numbers, letters, and some form of punctuation. Using a password based on an acronym is easier to remember (i.e. "I like to eat vanilla ice cream" could equate to "Il2evic").

b) ___ Add at least one non-root user

Make it a practice not to log into the system as root and don't `su` to root except when performing required administrative tasks. Once done, drop back down to regular user level.

1.11) Authentication Configuration

Go with the defaults.

a) ___ Enable md5 passwords (yes)
b) ___ Enable shadow password (yes)
c) ___ Do not enable NIS.

1.12) Selecting Package Groups

a) ___ Development (this will be removed after our system is fully installed)
b) ___ Utilities
c) ___ Mail/WWW/News Tools
d) ___ Networked workstation

1.13) ___ Check 'Select individual packages'

Getting rid non-essentials is a good step toward a more secure system. On the other hand, there are a few handy tools that need to be selected. So, we go through it all, package by package.

Individual selections and removals:

a) ___(Add) Applications->Archiving: unzip, zip
b) ___(Add) Applications->Editors: vim-enhanced
c) ___(Remove) Applications->Internet: elm, rsh, rsync
d) ___(Add) Applications->Internet: tcpdump, wget

e) **___(Remove) Applications->Publishing:** rhs-printfilters
f) **___(Remove) Applications->System:** rdate
g) **___(Add) Applications->System:** vlock
h) **___(Remove) System Environment->Base:** yp-tools
i) **___(Remove) System Environment->Daemons:** lpr, nfs-utils, pidentd, portmap, rusers, rwho, ypbind
j) **___(Add) System Environment->Daemons:** inetd, xntp3

When done with the selections, click 'Next'.

About to Install

Click 'Next' to install the selected packages. Then go get a cup of coffee and wait for the package installation process to complete.

1.14) ___ Boot disk creation

Do it, label it appropriately, and store it in a safe place. The boot disk can be used in an emergency to boot your system should the hard drive fail to boot for whatever reason. This sometimes happens when one updates the kernel without updating /etc/lilo.conf and running LILO.

Exit the installer.

Click 'Exit'. The system will now eject the installation cd-rom and reboot. If you booted with a boot floppy, you will need to eject it at this time. Remove the cd-rom or floppy from the drive to ensure that the system boots from the hard disk.

Step 2:
Updating the base installation

Before plugging in to the network in order to download updates, we need to make sure we don't have services unnecessarily listening for connections. It would not be good to be compromised by some lucky attacker while in the process of downloading the security updates and bug fixes.

2.1) ___ Login as the user account you created. Then `su` to root:

```
[~user]$ /bin/su -
Password:
[/root]#
```

Notice that the full path to `su` is specified. This will lessen the likelihood of someone placing their own version of `su` in your path and having you unwittingly execute it. The `su` command is particularly enticing to attackers since the root password is passed to it.

2.2) ___ Comment out all services in `/etc/inetd.conf`

Use `vi` or `emacs` to edit `/etc/inetd.conf`, and wherever you see a service that is not commented out, place a '#' as the first character on the line. For example, the ftp entry looks like this:

```
ftp  stream    tcp  nowait    root /usr/sbin/tcpd in.ftpd –l –a
```

Change it to:

```
#ftp stream    tcp  nowait    root /usr/sbin/tcpd in.ftpd –l –a
```

After commenting out all the services, save and exit your edit session.

2.3) ___ Set `/etc/inetd.conf` immutable

```
[/root]# chattr +i /etc/inetd.conf
```

According to the `chattr` man page:

"`A file with the `i' attribute cannot be modified: it cannot be deleted or renamed, no link can be created to this file and no data can be written to the file. Only the super user can set or clear this attribute.`"

2.4) ___ Turn off inet for now

Since all services are commented out of `/etc/inetd.conf`, go ahead and turn off inet:

```
[/root]# /etc/rc.d/init.d/inet stop
[/root]# chkconfig inet off
```

You can always turn inet back on again if it is needed at some later time.

2.5) Find out what services or daemons are still listening for connections:

a) ___ `[/root]# netstat –at`

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 *:smtp                  *:*                     LISTEN
```

You should see something similar to the above.

2.6) Disable sendmail daemon mode

a) ___ Stop sendmail

```
[/root]# /etc/rc.d/init.d/sendmail stop
```

b) ___ Edit `/etc/sysconfig/sendmail` to read:

```
DAEMON=no
QUEUE=10m
```

This tells sendmail to flush the outgoing mail queue every 10 minutes, but it will not be listening on port 25 for incoming connections. Generally mail will be sent out immediately, but if the destination mail server is temporarily unavailable, sendmail will queue the mail. This will ensure that sendmail continues to attempt mail delivery until it succeeds or fails permanently (usually after trying for 5 days).

c) ___ Start sendmail

```
[/root]# /etc/rc.d/init.d/sendmail start
```

d) ___ Run `netstat` again to verify that sendmail isn't listening

```
[/root]# netstat –at

Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
```

Okay, now that we don't have any services listening, we should be okay to connect to the network and download package updates.

e) ___ Connect network cable to NIC.

Verify that you have a link light showing on your NIC (if applicable).

f) ___ Bounce the interface and set default gateway

Since we booted without an active network connection, the Ethernet interface will likely need to be "bounced" (brought down and then back up). You will also need to enter the default gateway into the routing table:

```
[/root]# ifconfig eth0 down
[/root]# ifconfig eth0 <ip-address> netmask <netmask> up
[/root]# route add default gw <gateway-ip>
```

Ensure that you can ping your gateway. If 192.168.100.1 is your gateway:

```
[/root]# ping -c 3 -n 192.168.100.1
PING 192.168.100.1 (192.168.100.1) from 192.168.100.12 : 56(84) bytes of data
64 bytes from 192.168.100.1: icmp_seq=0 ttl=255 time=719 usec
64 bytes from 192.168.100.1: icmp_seq=1 ttl=255 time=542 usec
64 bytes from 192.168.100.1: icmp_seq=2 ttl=255 time=491 usec

--- 192.168.100.1 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/mdev = 0.491/0.584/0.719/0.097 ms
```

The network is up.

2.7) Download the updated RPM packages

A few Redhat update mirror sites are listed below:

ftp-linux.cc.gatech.edu/pub/Linux/distributions/redhat/updates/6.2/i386
acs-mirror.ucsd.edu/linux/redhat/updates/6.2/i386
ftp.cis.ohio-state.edu/mirror/redhat-updates/6.2/i386

More update mirror sites can be found at http://www.redhat.com/mirror.html.

a) ___ Create directories to store updates in

```
[/root]# mkdir /usr/local/updates
[/root]# mkdir /usr/local/updates/kernel
[/root]# cd /usr/local/updates
```

b) ___ Download the updates from one of the Redhat mirror sites

```
[/usr/local/updates]# ncftp \
ftp://ftp.cis.ohio-state.edu/mirror/redhat-updates/6.2/i386

ncftp> get *.rpm
```

This could take a LONG time depending on the speed of your network connection. In some cases it may be a few minutes and in others it may be several hours. All in all we're looking at approximately 140MB of packages to download.

```
ncftp> quit
```

2.8) Apply the updates

First, move the kernel packages into the separate directory we made earlier:

a) ___ Move kernel updates into separate directory

```
[/usr/local/updates]# mv kernel-* kernel/
```

b) ___ Apply updates with the "freshen" option

```
[/usr/local/updates]# rpm -Fvh *.rpm --nodeps
```

The `-F` option stands for "freshen". This tells RPM to only update packages that are currently installed on the system. The `--nodeps` option tells RPM to update the packages without doing a dependency check. Without this option, you would be required to install many of the packages in a particular order to satisfy certain packages' dependencies on other packages, which can be very cumbersome. Additionally, some packages have circular dependencies which can't be resolved without resorting to a `--nodeps` switch anyway.

Install the new kernel updates with the `-ivh` option to RPM. This keeps RPM from replacing the old kernel – just in case we have problems with the new kernel version and need to fall back on the old one.

c) ___ Apply the kernel updates

```
[/usr/local/updates]# cd kernel
[…/updates/kernel]# rpm -ivh kernel-2.2.16-3.i386.rpm
[…/updates/kernel]# rpm -ivh kernel-BOOT-2.2.16-3.i386.rpm
[…/updates/kernel]# rpm -ivh kernel-headers-2.2.16-3.i386.rpm
[…/updates/kernel]# rpm -ivh kernel-source-2.2.16-3.i386.rpm
```

Remove `kernel-pcmcia-cs` and old `kernel-utils` packages. The `kernel-pcmcia-cs` package is not needed, and `kernel-utils` doesn't like it when we try to install the newer package:

```
[…/updates/kernel]# rpm -e kernel-pcmcia-cs
[…/updates/kernel]# rpm -e kernel-utils
```

Now add the new kernel-utils package:

```
[…/updates/kernel]# rpm -ivh kernel-utils-2.2.16-3.i386.rpm
```

d) ___ Edit `/etc/lilo.conf`

Edit the "image" sections of `/etc/lilo.conf` to reflect the following:

```
image=/boot/vmlinuz
     label=linux
     read-only
     root=/dev/hda1

image=/boot/vmlinuz-2.2.14-5.0
     label=linux.old
     read-only
     root=/dev/hda1
```

e) ___ Run LILO to write the changes to disk

```
[/etc]# /sbin/lilo
Added linux *
Added linux.old
```

d) ___ Reboot

```
[/etc]# reboot
```

If everything has gone smoothly, your system should now boot using the newly installed 2.2.16 kernel. Once the login prompt appears, login and `su` to the root user for further installation and configuration.

Step 3:
Install additional software

3.1) Install OpenSSL and OpenSSH

Ssh will be the only method of accessing the server from a remote (non-local console) location. OpenSSH was chosen over other SSH implementations because it is free and now has SSH2 support. OpenSSH requires that OpenSSL be installed on the system. I've had better luck compiling these packages from scratch as opposed to using the pre-compiled binary RPM packages. The latest versions of OpenSSH and OpenSSL as of this writing are 2.3.0p1 and 0.9.6, respectively. Be sure to download the latest versions of each:

a) ___ Download OpenSSL

```
[/root]# mkdir /usr/local/build
[/root]# cd /usr/local/build
[…/build]# ncftp ftp://ftp.openssl.org/source
ncftp /source> get openssl-0.9.6.tar.gz
```

b) ___ Download OpenSSH

```
[…/build]# ncftp ftp://ftp.openbsd.org/pub/OpenBSD/OpenSSH/portable
ncftp …/portable> get openssh-2.3.0p1.tar.gz
```

c) ___ Compile & install OpenSSL

```
[…/build]# tar zxvf openssl-0.9.6.tar.gz
[…/build]# cd openssl-0.9.6
[…/build/openssl-0.9.6]# ./config ; make ; make install
[…/build/openssl-0.9.6]# cd ..
```

d) ___ Compile & install OpenSSH

```
[…/build]# tar zxvf openssh-2.3.0p1.tar.gz
[…/build]# cd openssh-2.3.0p1
[…/build/openssh-2.3.0p1]# ./configure --sysconfdir=/etc/ssh
[…/build/openssh-2.3.0p1]# make ; make install
[…/build/openssh-2.3.0p1]# cp contrib/redhat/sshd.pam /etc/pam.d/sshd
```

e) ___ Edit `/etc/ssh/sshd_config` to disallow root login via SSH

Change "`PermitRootLogin yes`" to "`PermitRootLogin no`".

f) ___ Edit `/etc/ssh/sshd_config` to enable sftp and scp

Uncomment the sftp subsytem at the bottom of the config file by removing the "`#`" from the beginning of the line. Change:

```
#Subsystem       sftp    /usr/local/libexec/sftp-server
```

to:

```
Subsystem       sftp    /usr/local/libexec/sftp-server
```

g) ___ Create a boot script to start sshd during system boot-up

Create the script listed below using your favorite editor:

```
[/root]# cd /etc/rc.d/init.d
[/etc/rc.d/init.d]# vi sshd

#!/bin/bash
#
#     /etc/rc.d/init.d/sshd
```

```
#
# sshd      Start the Secure Shell daemon
#
# chkconfig: 345 55 55
# description:    The Secure Shell daemon, sshd, allows for strong \
#                 authentication of, and encrypted communications with, \
#                 remote clients using the Secure Shell Protocol
# processname:    sshd
# pidfile:        /var/run/sshd.pid
# config:         /etc/ssh/sshd_config

# Source function library
. /etc/rc.d/init.d/functions

SSHD=/usr/local/sbin/sshd
SSHD_CONFIG=/etc/ssh/sshd_config

case "$1" in
      start)
            echo -n "Starting SSH services: "
            if [ -x $SSHD -a -f $SSHD_CONFIG ]
            then
                  daemon $SSHD
            else
                  echo_failure
            fi
            echo
            touch /var/lock/subsys/sshd
            ;;
      stop)
            echo -n "Shutting down SSH services: "
            killproc sshd
            echo
            rm -f /var/lock/subsys/sshd
            ;;
      status)
            status sshd
            ;;
      restart)
            $0 stop; $0 start
            ;;
      reload)
            killall -HUP sshd
            ;;
      *)
            echo "Usage: sshd {start|stop|status|restart|reload}"
            exit 1
            ;;
esac
```

Now make it executable and use `chkconfig` to create the symbolic links in the appropriate run-level directories:

```
[/etc/rc.d/init.d]# chmod +x sshd
[/etc/rc.d/init.d]# chkconfig sshd on
```

h) ___ Create symbolic links for "r" programs to the ssh programs

```
[/root]# ln –s /usr/local/bin/ssh /usr/bin/rsh
[/root]# ln –s /usr/local/bin/slogin /usr/bin/rlogin
[/root]# ln –s /usr/local/bin/scp /usr/bin/rcp
```

3.2) Download and install AIDE

AIDE (Advanced Intrusion Detection Environment) is a free replacement for the Tripwire program. Similar to Tripwire, AIDE creates a database of cryptographic hashes for a pre-defined set of files. This database is then used for occasional integrity checking to insure that those files have not been modified in any way. One of the best ways to detect and understand the extent of an intrusion is to discover that certain files have been modified and, more importantly, to know which ones.

You can read more about AIDE at the author's website:
http://www.cs.tut.fi/~rammer/aide.html

a) ___ Download AIDE

```
[/root]# cd /usr/local/build
[…/build]# ncftp ftp://ftp.cs.tut.fi/pub/src/gnu
ncftp …/gnu> get aide-0.7.tar.gz
```

b) ___ Compile & install AIDE

```
[…/build]# tar zxvf aide-0.7.tar.gz
[…/build]# cd aide-0.7
[…/build/aide-0.7]# ./configure
[…/build/aide-0.7]# make ; make install
```

Since we still have to make some changes on the system we will not configure AIDE at this time. Creating the initial AIDE database will be the last thing we do before making our initial backup and placing the server into production.

Step 4:
Securing and optimizing the system

The following series of steps will take you all over your system, editing various files, tweaking settings, and generally fixing things that are "broken" in the default install.

4.1) Fix `/etc/inittab`

a) ___ Disable "Control-Alt-Delete" rebooting

change:

```
ca::ctrlaltdel:/sbin/shutdown -t3 -r now
```

to:

```
#ca::ctrlaltdel:/sbin/shutdown -t3 -r now
```

b) ___ Require root password when booting to single user mode

add:

```
~~:S:wait:/sbin/login
```

after the entry for "`si::sysinit:/etc/rc.d/rc.sysinit`."

c) ___ Set `/etc/inittab` immutable

```
[/root]# chattr +i /etc/inittab
```

4.2) Password protect LILO command-line options

a) ___ Edit `/etc/lilo.conf` to include the following after the `prompt` entry:

```
password = good-password
restricted
```

Replace "good-password" with your own password.

Warning: Be sure not to forget the `restricted` option. This relaxes the protection by requiring a password only if parameters are specified on the command line. If you don't specify the `restricted` option, your server will fail to come back online in the event of a power loss. Once power is restored the boot process will hang at the point where it prompts for the password in the console.

b) ___ Set permissions for `/etc/lilo.conf`

```
[/root]# chmod 600 /etc/lilo.conf
[/root]# chattr +i /etc/lilo.conf
```

c) ___ Run LILO to write changes

```
[/root]# /sbin/lilo
```

```
Added linux *
Added linux.old
```

4.3) Run the "newperms" script

Download and run the "newperms" script that changes the permissions on potentially sensitive files so that regular users can't read, write, or execute them:

```
[/root]# wget www.sans.org/linux/newperms
[/root]# sh newperms
```

You will see some "No such file or directory" error messages. These errors are due to the fact that we never installed the packages that contain those files.

4.4) Check for SUID and SGID programs

Most systems come with too many applications that are set SUID or SGID unnecessarily. Many of these SUID and SGID bits can be turned off without adversely affecting the intended operation of the application. In some cases you just have to run them as root.

a) ___ Find SUID and SGID applications

```
[/root]# find / \( -perm -004000 -o -perm -002000 \) -type f -exec ls -l {} \;

-rwsr-xr-x    1 root     root        35168 Feb 16  2000 /usr/bin/chage
-rwsr-xr-x    1 root     root        36756 Feb 16  2000 /usr/bin/gpasswd
-r-xr-sr-x    1 root     tty          6128 Mar  7  2000 /usr/bin/wall
-rwsr-xr-x    1 root     root        33288 Mar  1  2000 /usr/bin/at
-r-xr-sr-x    1 news     news        73144 Mar  2  2000 /usr/bin/inews
-rws--x--x    2 root     root       531612 Aug 10  2000 /usr/bin/suidperl
-rws--x--x    2 root     root       531612 Aug 10  2000 /usr/bin/sperl5.00503
-rwxr-sr-x    1 root     man         34800 Jun 30  2000 /usr/bin/man
-r-s--x--x    1 root     root        12244 Feb  7  2000 /usr/bin/passwd
-rwxr-sr-x    1 root     mail        11620 Feb  7  2000 /usr/bin/lockfile
-rwsr-sr-x    1 root     mail        76432 Feb  7  2000 /usr/bin/procmail
-rwxr-sr-x    1 root     slocate     20880 Dec 18 12:16 /usr/bin/slocate
-rws--x--x    1 root     root        14056 Mar  7  2000 /usr/bin/chfn
-rws--x--x    1 root     root        13832 Mar  7  2000 /usr/bin/chsh
-rws--x--x    1 root     root         5640 Mar  7  2000 /usr/bin/newgrp
-rwxr-sr-x    1 root     tty          8328 Mar  7  2000 /usr/bin/write
-rwsr-xr-x    1 root     root        21816 Feb  3  2000 /usr/bin/crontab
-rws--x--x    1 root     root       644404 Feb  8 19:37 /usr/local/bin/ssh
-rwsr-xr-x    1 root     root         5896 Mar  8  2000 /usr/sbin/usernetctl
-rwsr-sr-x    1 root     root       320516 Feb 17  2000 /usr/sbin/sendmail
-rwsr-xr-x    1 root     root        17672 Oct  4 15:31 /usr/sbin/traceroute
-rwxr-sr-x    1 root     utmp         6096 Feb 24  2000 /usr/sbin/utempter
-rwsr-xr-x    1 root     root        34751 Jan 15 10:50 /usr/libexec/pt_chown
-rwsr-xr-x    1 root     root        21072 Oct 10 16:37 /bin/ping
-rwsr-xr-x    1 root     root        14188 Mar  7  2000 /bin/su
-rwsr-xr-x    1 root     root        56208 Feb  3  2000 /bin/mount
```

```
-rwsr-xr-x    1 root     root        26608 Feb  3  2000 /bin/umount
-rwxr-sr-x    1 root     root         3860 Mar  8  2000 /sbin/netreport
-r-sr-xr-x    1 root     root        15688 Nov 30 16:02 /sbin/pwdb_chkpwd
-r-sr-xr-x    1 root     root        16312 Nov 30 16:02 /sbin/unix_chkpwd
```

Many of these are unnecessarily set SUID and SGID.

b) ___ Turn off SUID and SGID bits on some applications

```
[/root]# chmod a-s /usr/bin/chage /usr/bin/gpasswd /usr/bin/wall
[/root]# chmod a-s /usr/bin/at /usr/bin/inews /usr/bin/lockfile
[/root]# chmod a-s /usr/bin/procmail /usr/bin/chfn /usr/bin/chsh
[/root]# chmod a-s /usr/bin/newgrp /usr/bin/write
[/root]# chmod a-s /usr/sbin/usernetctl /usr/libexec/pt_chown
[/root]# chmod a-s /bin/mount /bin/umount /sbin/netreport
```

You could also turn off the SUID bits for `traceroute` and `ping` if you don't want your users using those programs. I like to be able to use them from a non-root account, so I leave them on.

c) ___ Recheck for SUID and SGID files

```
[/root]# find / -type f \( -perm -004000 -o -perm -002000 \) -exec ls -l {} \;
-rws--x--x    2 root     root       531612 Aug 10  2000 /usr/bin/suidperl
-rws--x--x    2 root     root       531612 Aug 10  2000 /usr/bin/sperl5.00503
-rwxr-sr-x    1 root     man         34800 Jun 30  2000 /usr/bin/man
-r-s--x--x    1 root     root        12244 Feb  7  2000 /usr/bin/passwd
-rwxr-sr-x    1 root     slocate     20880 Dec 18 12:16 /usr/bin/slocate
-rwsr-xr-x    1 root     root        21816 Feb  3  2000 /usr/bin/crontab
-rws--x--x    1 root     root       644404 Feb  8 19:37 /usr/local/bin/ssh
-rwsr-sr-x    1 root     root       320516 Feb 17  2000 /usr/sbin/sendmail
-rwsr-xr-x    1 root     root        17672 Oct  4 15:31 /usr/sbin/traceroute
-rwxr-sr-x    1 root     utmp         6096 Feb 24  2000 /usr/sbin/utempter
-rwsr-xr-x    1 root     root        21072 Oct 10 16:37 /bin/ping
-rwsr-xr-x    1 root     root        14188 Mar  7  2000 /bin/su
-r-sr-xr-x    1 root     root        15688 Nov 30 16:02 /sbin/pwdb_chkpwd
-r-sr-xr-x    1 root     root        16312 Nov 30 16:02 /sbin/unix_chkpwd
```

4.5) Enhance syslogd logging

a) ___ Edit `/etc/syslog.conf` and add the following:

```
*.warn;*.err   /var/log/syslog
kern.*         /var/log/kernel
```

b) ___ Touch new log files and restart syslogd

```
[/root]# touch /var/log/syslog /var/log/kernel
[/root]# chmod 700 /var/log/syslog /var/log/kernel
```

```
[/root]# killall -HUP syslogd
```

4.6) Configure log rotation for the new log files

a) ___ Edit `/etc/logrotate.d/syslog` and add:

```
/var/log/kernel {
     compress
     postrotate
          /usr/bin/killall -9 klogd
          /usr/bin/klogd &
     endscript
}

/var/log/syslog {
     compress
     postrotate
          /usr/bin/killall -HUP syslogd
     endscript
}
```

4.7) More configuration of log rotation

a) ___ Edit `/etc/logrotate.conf`

change:

```
# rotate log files weekly
weekly
```

to:

```
# rotate log files monthly
monthly
```

Also change:

```
# keep 4 weeks worth of backlogs
rotate 4
```

to:

```
# keep a year worth of backlogs
rotate 12
```

Also uncomment the "compress" option:

```
# uncomment this if you want your log files compressed
compress
```

4.8) Configure xntpd

a) ___ Edit `/etc/ntp.conf` and add the following lines:

```
server 132.239.254.5          # timekeeper@ucsd.edu
server 128.175.2.33           # Dave Mills mills@udel.edu
server 130.126.24.53          # Charley Kline kline@uiuc.edu
server 128.59.16.20           # timekeeper@cs.columbia.edu

restrict 132.239.254.5   nomodify noquery
restrict 128.175.2.33    nomodify noquery
restrict 130.126.24.53   nomodify noquery
restrict 128.59.16.20    nomodify noquery
restrict 127.0.0.1       nomodify
```

The time servers listed above are "open access" stratum 2 servers. However, it is considered good form to contact the administrators of the time servers to let them know you are synchronizing to them. The proper contacts are commented in the listing above. If you prefer to choose a different set of time servers, you can find the list of current stratum 2 time servers at:
http://www.eecis.udel.edu/~mills/ntp/clock2.htm

(Failing to contact the ntp admins when synchronizing to their time servers can earn you the ill-famed title of "clock sucker".)

b) ___ Start xntpd

```
[/root]# /etc/rc.d/init.d/xntpd start
```

c) ___ Set xntpd to start during system boot-up

```
[/root]# chkconfig xntpd on
```

4.9) ___ Turn off extraneous daemons

```
[/root]# chkconfig xfs off
[/root]# chkconfig atd off
[/root]# chkconfig gpm off
[/root]# chkconfig netfs off
[/root]# chkconfig kudzu off
[/root]# chkconfig linuxconf off
```

4.10) Add `/usr/local/bin` and `/usr/local/sbin` to the default PATH

a) ___ Edit `/etc/profile` and change "PATH" as follows:

```
PATH="$PATH:/usr/local/sbin:/usr/local/bin"
```

4.11) Remove compilers

Since we are done compiling programs for the initial install, let's remove the compiler packages to make it more difficult for potential attackers to build their tools locally. We can easily re-install them if we need to:

a) ___ Remove egcs packages

```
[/root]# rpm -e egcs-c++-1.1.2
[/root]# rpm -e cpp-1.1.2 --nodeps
[/root]# rpm -e egcs-1.1.2
```

4.12) IP stack tuning

The following IP stack optimizations are from Rob Thomas' UNIX IP Stack Tuning Guide v2.7 (http://www.cymru.com/~robt/Docs/Articles/ip-stack-tuning.html).

a) ___ Add the following entries to `/etc/sysctl.conf`:

```
# Socket queue defense against SYN attacks
net.ipv4.tcp_max_syn_backlog=1280
net.ipv4.tcp_syn_cookies=1

#Disable IP redirects
net.ipv4.conf.all.send_redirects=0
net.ipv4.conf.all.accept_redirects=0

#Drop source routed packets
net.ipv4.conf.all.accept_source_route=0

#Don't forward source routed frames
net.ipv4.conf.all.forwarding=0
net.ipv4.conf.all.mc_forwarding=0

#Prevent sockets from lingering in TIME_WAIT state
net.ipv4.vs.timeout_timewait=60

#Don't respond to directed broadcasts (SMURF)
net.ipv4.icmp_echo_ignore_broadcasts=1
```

4.13) Warning banners

It is a good idea to have warning banners that display before a user logs on or immediately after a user logs on. In many jurisdictions, their existence and proper display are statutory prerequisites to charging an intruder with criminal wrongdoing. Since banners are usually very site-specific, and often require the review of legal counsel, we will not cover the specifics of how to implement banners in this guide. However, a good write-up on implementing banners can be found at the CIAC web site:

http://www.ciac.org/ciac/bulletins/j-043.shtml

Additionally, a good article that describes how to implement banners using TCP Wrappers can be found at:

http://www.linuxgazette.com/issue15/tcpd.html

4.14) Set some limits on the users

a) ___ Edit `/etc/security/limits.conf` **and add:**

```
*     hard      core      0          # no core files
*     soft      nproc     50         # default 50 processes max
*     hard      nproc     75         # extreme 75 processes max
*     hard      rss       5000       # max memory 5MB
```

The setting above are just a baseline. You can, of course, edit them to suit your needs. The idea is to make it difficult for a single user to effect a denial of service by consuming too many system resources.

The settings do NOT affect the root user.

b) ___ Edit `/etc/pam.d/login` **and add:**

```
session   required  /lib/security/pam_limits.so
```

4.15) Configure AIDE

Now that we have configured and optimized our system, we need to have AIDE build a database of digital "fingerprints" of certain files and directories. Once this database is created, AIDE will use it as a canonical source for future file comparison and integrity checks.

We will store our database on a write-protected floppy to prevent modification by an attacker.

Warning: be sure to configure your BIOS to boot from the hard drive before the floppy disk (or disable booting from the floppy drive altogether). Otherwise a reboot will cause your system to hang as it tries to boot from your AIDE database floppy. Another option would be to burn a CD-ROM disc with the AIDE database on it.

a) ___ Create `/etc/aide/aide.conf`

Using your favorite editor, create the aide.conf configuration file:

```
[/root]# mkdir /etc/aide
[/root]# cd /etc/aide
[/etc/aide]# vi aide.conf

# AIDE configuration file

database=file://localhost/mnt/floppy/aide.db
database_out=file://localhost/etc/aide/aide.db.new

# Directories and files to include in the database

/bin R
/sbin R
/usr/bin/w R
/usr/bin/who R
/usr/bin/last R
/usr/bin/find R
/usr/sbin/lsof R
/usr/local/sbin/sshd R
```

Now fix the permissions:

```
[/etc/aide]# chmod 400 aide.conf
[/etc/aide]# chattr +i aide.conf
```

b) ___ Build the AIDE database

```
[/etc/aide]# aide --init --config=/etc/aide/aide.conf
[/etc/aide]# chmod 600 aide.db.new
```

c) ___ Create an ext2 floppy and copy `aide.db` to it

Insert a new floppy disk into the floppy drive.

```
[/etc/aide]# fdformat /dev/fd0H1440
[/etc/aide]# mke2fs -c /dev/fd0
[/etc/aide]# mount /dev/fd0 /mnt/floppy
[/etc/aide]# cp aide.db.new /mnt/floppy/aide.db
[/etc/aide]# chmod 400 /mnt/floppy/aide.db
```

```
[/etc/aide]# chattr +i /mnt/floppy/aide.db
[/etc/aide]# umount /dev/fd0
```

For good measure, create a second floppy with the AIDE database on it and put it away for safekeeping.

d) ___ Write-protect AIDE database floppy and insert it into floppy drive

e) ___ Edit `/etc/fstab` so that the floppy is mounted on boot up

Edit the `/dev/fd0` portion of the `/etc/fstab` file to read:

```
/dev/fd0                /mnt/floppy             auto    owner,ro        0 0
```

f) ___ Create a crontab entry and script to run AIDE daily

AIDE integrity checks should be run daily and the output should be made available to those responsible for the security of the system. Create a crontab entry and a script that runs AIDE on a daily basis and mails the results to any concerned parties:

```
[/root]# cd /etc/aide
[/etc/aide]# vi aide-send

#!/bin/sh

DATE=`date`
TO="root"
DATA=`/usr/local/bin/aide --check --config=/etc/aide/aide.conf`
SUBJECT="Subject: Aide report - $DATE"
FOOTER="To modify this script, edit /etc/aide/aide-send"

if [ "$DATA" = "" ]; then
      DATA="The AIDE file integrity check returned no changes."
fi

echo -e "$SUBJECT\n\n$DATA\n\n$FOOTER" | /usr/sbin/sendmail $TO
```

Save the script and make it executable only by root:

```
[/etc/aide]# chmod 700 aide-send
```

Create a crontab entry:

```
[/etc/aide]# crontab -e

59 7 * * * /etc/aide/aide-send
```

Save and exit. Cron will now run AIDE every day at 7:59 a.m. and mail the results to the users specified in the aide-send script.

Step 5:
Initial backup

Since we've invested a large amount of time into this system, we want to have a backup handy in case of disaster. Implementing a backup system and policy is beyond the scope of this guide. However, if you have a spare hard drive (or maybe you are lucky and have a SCSI card and a SCSI tape drive connected to this server), we will create a backup that can be used until you develop a more permanent solution or you integrate this server into your organization's backup procedure. Be sure that the spare drive is as large or larger than your system's drive.

5.1) Backup MBR and copy the partition table

Before performing the backup, it is a good idea to have a copy of your hard drive's master boot record (MBR) and a printout of the partition table. Insert and format a new floppy and do the following:

a) ___ Copy MBR to floppy

```
[/root]# dd if=/dev/hda of=/mnt/floppy/MBR.hda count=1
```

b) ___ Copy partition layout

```
[/root]# fdisk -l /dev/hda > /mnt/floppy/PTBL.hda.txt
```

5.1) Connect a spare drive to the system

a) ___ Halt the system

```
[/root]# halt
```

b) ___ Connect spare drive to IDE bus

Generally the device assignments will work out like this:

Primary Master	/dev/hda
Primary Slave	/dev/hdb
Secondary Master	/dev/hdc
Secondary Slave	/dev/hdd

5.2) Reboot server and determine device assignment of new drive

If you keep a close eye on the boot messages as the system comes up, you should be able to see what device name gets assigned to the new drive. If for some reason you miss it, or it scrolls by too quickly, you can issue this command to find the information once you've logged in and changed to root:

```
[/root]# dmesg | grep hd
```

5.3) Backup the main drive to the spare drive

a) ___ Use `dd` to backup the entire disk

Assuming that your spare drive was designated /dev/hdb:

```
[/root]# dd if=/dev/hda of=/dev/hdb
```

b) ___ Run `fsck` on the spare drive's partitions

For each partition excluding any extended partitions or swap partitions, run:

```
[/root]# fsck –y /dev/hdb#
```

where # is the partition number.

Halt the system, remove the backup drive, and store for safekeeping. The system is now ready for use.

Conclusion

We now have a server that should be relatively secure. You can now bring the system online.

Be sure to maintain a vigilant attitude toward the security of the system: review logs on a daily basis and keep up with bug fixes and security updates. It would be a good idea to subscribe to the Redhat Announce mailing list where they post security and update announcements: http://www.redhat.com/mailing-lists/list_subscribe.html

While it is beyond the scope of this guide to cover packet filtering using ipchains and do the subject justice, if packet filtering is a security option you would like to pursue, check out http://www.linux-firewall-tools.com/linux. There is also a good ipchains firewalling script available at http://www.sans.org/linux.

Regarding backups and recovery in the Unix environment, an excellent source on the subject is "Unix Backup & Recovery" by W. Curtis Preston, published by O'Reilly and Associates.

References

Securing Linux Step-by-step, Version 1.0
(The SANS Institute)

SANS Securing Unix Systems, Linux Practicum
by Lee E. Brotzman (The SANS Institute)

Practical Unix and Internet Security
by Simson Garfinkel and Gene Spafford (O'Reily & Associates)

Unix Backup & Recovery
by W. Curtis Preston, et al (O'Reily & Associates)

LILO User's Guide
by Werner Almesberger