



# Global Information Assurance Certification Paper

Copyright SANS Institute  
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permitted without express written permission.

Interested in learning more?

Check out the list of upcoming events offering  
"Securing Linux/Unix (Security 506)"  
at <http://www.giac.org/registration/gcux>

**GIAC Level Two Securing Unix  
GCUX Practical Assignment  
Version 1.6d**

**Mike Dietz**

© SANS Institute 2000 - 2005, Author retains full rights.

## Building a secure Linux shell server

This document will outline the basic steps necessary to create a secure shell server using RedHat Linux 7.1. The server will offer shell accounts and email access to subscribers. The installation does not cover user software like mail clients, ftp clients, and other non-system related software. This server will be placed in a physically secured semi public data center. ID access to the data center is required, however multiple companies are using the co-location provider as an ISP and can physically access the box. If the machine is removed from the rack or disassembled it requires another security clearance that only the owner of the machine has. The main threat from a physical access standpoint is someone at the console attempting to insert or remove media, power off, or attempting to log in.

The initial number of users will be less than 500. The hardware requirements listed below are given with expansion in mind. We assume the users are Unix savvy and may be likely to attempt local and remote exploits. Attacks against the local machine or remote machines by local users will not be tolerated.

Before beginning the installation several precautions should be taken. The machine should be unplugged from the network and installed in a private physically secured location until completely hardened. All software to be installed on the machine including updates should be burned on a CD so that no network access is necessary during the hardening process. Following installation, several rigorous tests will be performed in order to insure the machine is properly secured.

The general strategy of this installation is defense in depth. Since this will be an isolated machine on a presumably hostile network, we will often do things that seem redundant or unnecessary to those unfamiliar with security. One example of such is disabling unnecessary services and creating a firewall rule set. It may appear that such processes are redundant, but in reality they complement each other by providing multiple layers an attacker must circumvent in order to compromise the system.

### Hardware and Physical Security

The rack mount box should have a lock or cage to prevent tampering with all power and reboot switches, cables, drives, ports, and any other external interfaces. The hardware requirements should be fairly minimal. Since current hardware is cheap the following setup is recommended:

- 1 GB RAM
- 2 1 GHz Processors
- 2 18 GB SCSI 10,000 RPM HD's
- 1 SCSI CD-ROM
- 1 Floppy Drive
- 1 10/100/1000 Ethernet card

The machine should be physically secured by doing the following, consult your hardware manual for the proper steps to perform this operation:

- \_\_\_\_\_ Disable booting from any removable media such as, floppy, CD-ROM, or tape in BIOS.
- \_\_\_\_\_ Password protect BIOS Setup with a unique alphanumeric password.
- \_\_\_\_\_ Disable booting from any removable media, floppy, CD-ROM, or tape in SCSI BIOS setup.
- \_\_\_\_\_ Password protect if possible the SCSI BIOS setup.

## Downloading Software and updates

Download all of the required software and updates needed beforehand on a secure machine, and burn them onto a CD-ROM. Several of the procedures require software to be compiled and configured, make sure you read the hardening instructions completely and copy the compiled binaries to the CD-ROM.

- \_\_\_\_\_ RedHat 7.1 updates from <ftp://updates.redhat.com/7.1/en/os/i386>
- \_\_\_\_\_ RedHat 7.1 powertools updates from <ftp://updates.redhat.com/7.1/en/powertools/i386/>
- \_\_\_\_\_ Latest OpenSSH from <http://www.openssh.com/portable.html>
- \_\_\_\_\_ Psionic Logcheck from <http://www.psionic.com/abacus/logcheck>
- \_\_\_\_\_ Linux Kernel from <ftp://ftp.kernel.org/pub/linux/kernel/v2.4>
- \_\_\_\_\_ Grsecurity Linux Kernel patch from <http://www.getrewted.net/>
- \_\_\_\_\_ Bastille Linux hardening script from <http://www.bastille-linux.org>

## Linux Installation

Make sure all network cables are unplugged before you begin the installation. You will temporarily have to allow booting from CD-ROM so you can install the RedHat 7.1 Operating System. Enter the SCSI firmware setup and select the option to boot from the CD-ROM. Insert the CD-ROM and reboot the machine to begin installation. When the initial prompt for what mode to install appears, type expert. Insert the driver diskette when prompted. Configure your SCSI adapter if it is not detected. Follow the below procedures:

- \_\_\_\_\_ Unplug network cables
- \_\_\_\_\_ Enable booting from SCSI CD-ROM
- \_\_\_\_\_ Insert CD-ROM and reboot
- \_\_\_\_\_ Type in expert mode
- \_\_\_\_\_ Configure SCSI adapter if not detected
- \_\_\_\_\_ Select English Language, US Keyboard
- \_\_\_\_\_ Install media is local CD-ROM
- \_\_\_\_\_ Select two Button Mouse
- \_\_\_\_\_ Choose custom System Install

## \_\_\_\_\_ Manually Partition with Disk Druid

The file system should be configured with multiple partitions, including separate /tmp, /var, and /var/spool/mail directories to limit possible exploits. A configuration similar to the following is recommended:

File System (Total space = 32 GB):		
/dev/sda1	/	2 GB
/dev/sda5	swap	2 GB
/dev/sda6	/tmp	2 GB
/dev/sda7	/var	2 GB
/dev/sda8	/var/spool/mail	10 GB
/dev/sdb1	/home	8 GB

- \_\_\_\_\_ Select Create Boot Disk
- \_\_\_\_\_ Install Lilo on the Master Boot Record
- \_\_\_\_\_ Do not select Linear mode unless you are sure your drive needs it
- \_\_\_\_\_ Configure your Ethernet card Eth0 with the parameters given by your ISP.
- \_\_\_\_\_ Choose No Firewall (RedHat installs an IP Chains Firewall, we will configure a netfilter firewall later)
- \_\_\_\_\_ Select Only the Language you use, for the US, Choose English, USA
- \_\_\_\_\_ Select the nearest city in your time zone
- \_\_\_\_\_ Choose the proper UTC offset
- \_\_\_\_\_ Setup the root account password
- \_\_\_\_\_ Enable MD5 passwords and shadow passwords
- \_\_\_\_\_ Deselect all Package groups and check Select individual packages
- \_\_\_\_\_ Under Applications Select the following packages:
  - \_\_\_\_\_ Archiving → dump
  - \_\_\_\_\_ Internet → openssh, openssh-clients
  - \_\_\_\_\_ System → bind-utils, mt-st, procinfo, symlinks, tripwire
  - \_\_\_\_\_ Text → M4, mawk, rgrep
- \_\_\_\_\_ Under Development Select the following packages
  - \_\_\_\_\_ Debuggers → lsk, lsof, ltrace, strace
- \_\_\_\_\_ Under Documentation Select the following packages
  - \_\_\_\_\_ man-pages, sendmail-doc
- \_\_\_\_\_ Under System Environment Select the following packages
  - \_\_\_\_\_ Base → iptables
  - \_\_\_\_\_ Daemons → ntp, openssh-server, sendmail-cf

After the installation is completed, enter the SCSI firmware setup and deselect the option to boot from the CD-ROM. Create an administrative group, a user account for yourself, and create a base home directory for users and staff:

- \_\_\_\_\_ /bin/mkdir /home/staff
- \_\_\_\_\_ /bin/mkdir /home/users
- \_\_\_\_\_ /usr/sbin/groupadd -g 1000 admin

```
_____ /usr/sbin/useradd -g admin -Gwheel -d /home/staff/sansid sansid  
_____ /usr/bin/passwd sansid
```

## Upgrading RPM Packages

After the installation, insert the CD-ROM that contains the latest updates you downloaded from <ftp://updates.redhat.com/7.1/en/os/i386>. Mount the CD-ROM with the latest updates, change to the directory you mounted the CD-ROM, and perform the update:

```
_____ /bin/rpm -Fvh *.rpm
```

RedHat doesn't always keep up with the latest OpenSSH updates. In some cases there may be minor security enhancements or features that are present in newer releases. Mount the CD-ROM containing the updates and update the RPMs by typing in:

```
_____ /bin/rpm -Fvh openssh*.rpm  
_____ Set more secure SSH daemon defaults, back up your current  
/etc/ssh/sshd_config, and edit the default to make it look as follows (comments removed):
```

```
Port 22  
HostKey /etc/ssh/ssh_host_key  
HostKey /etc/ssh/ssh_host_rsa_key  
HostKey /etc/ssh/ssh_host_dsa_key  
ServerKeyBits 768  
LoginGraceTime 600  
KeyRegenerationInterval 3600  
PermitRootLogin no  
IgnoreRhosts yes  
StrictModes yes  
X11Forwarding no  
PrintMotd yes  
KeepAlive no  
SyslogFacility AUTH  
LogLevel INFO  
RhostsAuthentication no  
RhostsRSAAuthentication no  
RSAAuthentication yes  
PermitEmptyPasswords no
```

Edit the /etc/pam.d/sshd file to use the RedHat /etc/pam.d/system-auth file, that we will add some enhancements to later. /etc/pam.d/sshd should look like this:

```
#%PAM-1.0
auth      required  /lib/security/pam_stack.so service=system-auth
account   required  /lib/security/pam_stack.so service=system-auth
password  required  /lib/security/pam_stack.so service=system-auth
session   required  /lib/security/pam_stack.so service=system-auth
```

The `/etc/hosts.allow` file comes with the TCP wrappers, and can be used to permit or deny hosts to specific services by IP address or DNS name, although it has many other uses. The Bastille hardening script below will add a default deny rule to the bottom of the file to deny any host from connecting to any service. The rules it inserts also mail an alert to the root account noting that a port-denial was encountered. The rules should look as follows (after Bastille has been run of course):

```
in.fingerd: ALL : ALLOW
ALL : ALL : spawn (/usr/sbin/safe_finger -l @%h | /bin/mail -s "Port Denial noted %d-%h" root) & : DENY
```

We must edit the `/etc/hosts.allow` file and add a line at the top of the file so that any host can connect via ssh from anywhere:

```
sshd: ALL : ALLOW
```

## Bastille Installation

Install Bastille, a Linux hardening tool from <http://www.bastille-linux.org>. This tool automates many Linux hardening procedures and gives very detailed descriptions of each option. Each item should be read thoroughly to make sure that the installer understands the implications. I cannot stress enough how many questions reading the descriptions may answer for the installer. This tool includes steps such as removing the SUID bit from several executables to prevent possible exploits, deactivating unnecessary services, etc.

```
_____ /bin/rpm -ivh perl-Curses-1.05-2mdk.i586.rpm
_____ /bin/rpm -ivh --nodeps Bastille-1.2.0-1.1mdk.noarch.rpm Bastille-Curses-
module-1.2.0-1.1mdk.noarch.rpm
```

Run the Bastille hardening script by typing in: `/usr/sbin/InteractiveBastille` . Now configure Bastille as follows:

```
_____ Firewall.pm module: No, we will create our own IPTables script later.
_____ disable SUID status for mount/umount: Yes
_____ disable SUID status for ping: Yes
_____ disable SUID status for at: Yes
_____ disable SUID status for usernetctl: Yes
_____ enforce password aging: Yes
_____ restrict cron to administrative accounts: Yes
```

- \_\_\_\_\_ allow root to login on tty's 1-6: No
- \_\_\_\_\_ password-protect the LILO prompt: Yes
- \_\_\_\_\_ reduce the LILO delay time to zero: Yes
- \_\_\_\_\_ boot from your hard drive: Yes
- \_\_\_\_\_ write the LILO changes to a boot floppy: No
- \_\_\_\_\_ disable CTRL-ALT-DELETE rebooting: Yes
- \_\_\_\_\_ password protect single-user mode: Yes
- \_\_\_\_\_ default-deny on TCP Wrappers and xinetd: Yes
- \_\_\_\_\_ deactivate telnet: Yes
- \_\_\_\_\_ deactivate ftp: Yes
- \_\_\_\_\_ put limits on system resource usage: Yes
- \_\_\_\_\_ restrict console access to a small group of user accounts: Yes
- \_\_\_\_\_ add additional logging: Yes
- \_\_\_\_\_ have a remote logging host: No
- \_\_\_\_\_ set up process accounting: No
- \_\_\_\_\_ disable apmd: Yes
- \_\_\_\_\_ disable GPM: Yes
- \_\_\_\_\_ deactivate the routing daemons: Yes
- \_\_\_\_\_ leave sendmail running in daemon mode: Yes
- \_\_\_\_\_ run sendmail via cron to process the queue: No
- \_\_\_\_\_ disable the VRFY and EXPN sendmail commands: Yes
- \_\_\_\_\_ install TMPDIR/TMP scripts: No, can cause ssh sessions to hang on exit.
- \_\_\_\_\_ Are you finished answering the questions: Yes

Reboot the machine to verify Bastille hardening was successful. The script can be re-executed again at anytime to re-harden the system.

## Psionic Logcheck

Next we compile and install Psionic Logcheck, you must compile this on another machine. This will install the files on the machine you compiled it on. If you do not want the files installed on the compiler machine, skip running make linux and instead just type: gcc -o ./src/logtail ./src/logtail.c

- \_\_\_\_\_ untar the file: tar -xzf logcheck-1.1.1.tar.gz
- \_\_\_\_\_ cd logcheck-1.1.1
- \_\_\_\_\_ make linux
- \_\_\_\_\_ cd ..;tar -czf logcheck-1.1.1.compiled.tar.gz

Copy the file to a floppy or other media and transfer it to the machine to be hardened. Edit the Makefile to prevent it from failing on the compile step:

- \_\_\_\_\_ untar the file you transferred: tar -xzf logcheck-1.1.1.compiled.tar.gz
- \_\_\_\_\_ cd logcheck-1.1.1
- \_\_\_\_\_ edit the Makefile and comment out this line:  
# \$(CC) \$(CFLAGS) -o ./src/logtail ./src/logtail.c



\_\_\_\_\_ make linux

Edit the /usr/local/etc/logcheck.sh to insert an email address to send the log reports to:

\_\_\_\_\_ SYSADMIN=sansid

add another log to be monitored under the line:

```
$LOGTAIL /var/log/maillog >> $TMPDIR/check.$$
```

Insert the lines:

```
_____ $LOGTAIL /var/log/syslog >> $TMPDIR/check.$$
```

```
_____ $LOGTAIL /var/log/loginlog >> $TMPDIR/check.$$
```

```
_____ $LOGTAIL /var/log/kernel >> $TMPDIR/check.$$
```

Logcheck now monitors the following files: /var/log/messages, /var/log/secure, /var/log/maillog, /var/log/syslog, /var/log/loginlog, /var/log/kernel. The Bastille script above configures the syslog configuration for optimal logging. See the appendix A for details on how the Bastille script configures the /etc/syslog.conf file. Edit the Logcheck files accordingly to weed out frequent messages that are not important using the following files:

\_\_\_\_\_ /usr/local/etc/logcheck.hacking -Reports on known messages from hack attacks,

it will send mail with the subject: ACTIVE SYSTEM ATTACK.

\_\_\_\_\_ /usr/local/etc/logcheck.violations -Reports on events that could indicate a security violation and includes keywords such as denied, failed, su, etc., it will send mail with the heading "Security Violations".

\_\_\_\_\_ /usr/local/etc/logcheck.violations.ignore -Filters against the logcheck.violations file, if the keyword is found the event is ignored and not reported as a violation.

It

may be reported as "Unusual System Activity".

\_\_\_\_\_ /usr/local/etc/logcheck.ignore -Anything that does not match the keywords in this file is reported as "Unusual System Activity".

For example, to ignore ntp update alerts, add this line to /usr/local/etc/logcheck.ignore:

```
ntpd.*: kernel pll status change
```

Edit the crontab file to make logcheck run every 10 minutes. Add this line to /etc/crontab:

```
_____ 0,10,20,30,40,50 * * * * root /usr/local/etc/logcheck.sh
```

Be aware that running the script every 10 minutes will generate a lot of error messages until you fine tune the ignore files to get rid of extraneous data. If the volume of data being reported after tuning is still too large to handle at the frequency of every 10 minutes, I would suggest reducing the checking period to every hour or so.

Logcheck handles the default RedHat logrotate program log rotation with no issues, so no further configuration needs to be done regarding log rotation. Logs are rotated weekly by default and saved for 4 rotations, and this should be sufficient for our needs. Logs should be archived to tape along with the rest of the data being backed up daily, however that is beyond the scope of this document. If your site accumulates large amounts of data quickly you may want to edit /etc/logrotate.conf and uncomment the line with #compress in it.

## Configure NTP

Pick three public stratum 2 servers from a location near you in your time zone from <http://www.eecis.udel.edu/~mills/ntp/servers.htm>. Ask permission from the administrator at each site prior to using the server to synchronize your time.

Edit the /etc/ntp.conf to configure the servers and restrict other machines from synchronizing time with this machine. Your configuration file you look something like the following:

```
# Stores clock drift data
driftfile /etc/ntp/drift
# pseudo clock, local
server 127.127.1.0          # local clock
fudge 127.127.1.0 stratum 10
# Remote stratum 2 servers
server 130.126.24.53      # ntp-0.cso.uiuc.edu
server 140.221.9.6        # ntp-2.mcs.anl.gov
server 128.105.39.11      # ntp1.cs.wisc.edu
# Set default to ignore ntp requests from other machines
restrict default         ignore
# Servers we synchronize cannot query/modify our local time
restrict 130.126.24.53   nomodify noquery
restrict 140.221.9.6     nomodify noquery
restrict 128.105.39.11   nomodify noquery
```

Make sure ntpd starts up each time the system restarts:

```
_____ /sbin/chkconfig ntpd on
```

## Sendmail

RedHat 7.1 by default configures sendmail to only listen to local connections. Since this machine needs to both send and receive email we need to enable sendmail to

listen to all interfaces. It does not allow relaying from other hosts by default and we are not going to allow relaying from other hosts. The sendmail greeting message should be changed to something less revealing.

In /etc/sendmail.cf

```
_____ Comment out:#O DaemonPortOptions=Port=smtp,Addr=127.0.0.1, Name=MTA
_____ add line under above: O DaemonPortOptions=Name=MTA
_____ Comment out: #O SmtgGreetingMessage=$j Sendmail $v/$Z; $b
_____ add line under above: O SmtgGreetingMessage=Mail Server Ready
_____ edit /etc/mail/local-host-names and add any aliases for your machine.
```

## Compile the Linux Kernel with enhanced security

It is necessary to compile a custom kernel in order to get features like a non-executable stack and Proc restrictions. Since many of the current attack methods rely on buffer overflows, non-executable stack protection can significantly slow down the average attacker because it prevents executing code after the buffer is overflowed. Complete instructions for compiling a custom kernel vary for each hardware type and will not be covered. See <http://www.linuxdoc.org/HOWTO/Kernel-HOWTO.html> for detailed instructions on how to compile the kernel for a specific hardware type. Unpack the latest kernel and copy the getrewted security patch to the linux directory. Then apply the kernel patch and use the menuconfig from the linux kernel to edit the kernel options.

```
_____ tar -xzf linux-2.4.6.tar.gz
_____ cp grsecurity-1.5-2.4.6.patch linux
_____ patch -p0 <grsecurity-1.5-2.4.6.patch
_____ make menuconfig
```

Using the menuconfig we need to enable iptables support. Under Networking Options select IP: Netfilter Configuration. Enable the following items as modules:

```
_____ Connection tracking (Needed for stateful filtering)
_____ FTP protocol support (Used for tracking FTP connections)
_____ IP tables support (Base component needed for firewalling)
_____ limit match support (Allows limiting rate of packet transfer)
_____ MAC address match support (Allows to filter by MAC address in addition to IP)
_____ Multiple port match support (Allows putting multiple ports in a rule)
_____ Connection state match support (Needed for stateful filtering)
_____ Packet filtering (Base component needed for firewalling)
_____ LOG target support (Ability to log packets to syslog facility)
```

Configure the getrewted security patch by navigating to the Getrewted Kernel Security section. The getrewted security patch supports a number of options that make it much more difficult to compromise a system. Enable the following options:

- \_\_\_\_\_ Getrewted Kernel Security
- \_\_\_\_\_ Openwall non-executable stack (Make buffer overflow exploits difficult)
- \_\_\_\_\_ Gcc trampoline support (Needed for glibc 2.0)
- \_\_\_\_\_ Proc restrictions (Tightens security on /proc)
- \_\_\_\_\_ Restrict to user only (Restricts users to seeing only processes they own)
- \_\_\_\_\_ Linking restrictions (Restricts linking to files you do not own in world writeable and sticky directories)
- \_\_\_\_\_ FIFO restrictions (Prevents users from creating FIFOs in directories they don't own or with the sticky bit set)
- \_\_\_\_\_ Secure file descriptors (Prevent data spoofing attacks using set\*id binaries)
- \_\_\_\_\_ Exec process limiting (Imposes user resource limits on execve() calls)
- \_\_\_\_\_ Signal logging (Log signals such as SIGSEGV)
- \_\_\_\_\_ Secure keymap loading (Prevents keyboard binding modification by unprivileged users)
- \_\_\_\_\_ Randomized PIDs (Randomizes process Ids)
- \_\_\_\_\_ Randomized IP Ids (Randomizes IP id field on outgoing packets)
- \_\_\_\_\_ Randomized TCP source ports (Makes new connection source ports less predictable)
- \_\_\_\_\_ Enhanced network randomness (Increases randomness of Linux IP stack)

## PAM Configuration

RedHat Linux uses Pluggable Authentication Modules (PAM) which allows one to add or change the method the operating system authenticates users. This allows us to limit failed logins, limit who can su, limit times users can log in, limit who can log in, and many other things which we won't discuss.

To prevent brute force attacks we are going to limit the number of failed login attempts to 10, after this the account will be prevented from logging in until the pam\_tally utility is ran to unlock the account. The number of failed logins will be reset after a successful login unless the account is locked from failed attempts. Some daemons such as the RedHat pop3 daemon do not work well with pam\_tally so we will back up the original configuration. All of the daemons we utilize work with pam\_tally however.

- \_\_\_\_\_ cp /etc/pam.d/system-auth /etc/pam.d/system-auth-default
- \_\_\_\_\_ edit /etc/pam.d/system-auth so it looks like this:

```
##%PAM-1.0
auth      required      /lib/security/pam_env.so
auth      required      /lib/security/pam_tally.so no_magic_root
auth      sufficient    /lib/security/pam_unix.so likeauth nullok
auth      required      /lib/security/pam_deny.so
```

```

reset
    account    required  /lib/security/pam_tally.so no_magic_root deny=10
    account    required  /lib/security/pam_unix.so

    password   required  /lib/security/pam_cracklib.so retry=3
    password   sufficient /lib/security/pam_unix.so nullok use_authtok md5
shadow
    password   required  /lib/security/pam_deny.so

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

```

```

_____ /bin/touch /var/log/faillog
_____ /sbin/pam_tally (to view or reset failed logins)

```

The utility `lastb` allows you to view the failed logins, and is similar to the `last` command except that it only displays failed logins. Unlike the `pam_tally` utility it allows you to view all failed logins since the last file rotation instead of just the last failed logins since the tally has been reset.

```

_____ /bin/touch /var/log/btmp
_____ /usr/bin/lastb (to view the failed logins)

```

Only allow members of group `wheel` to `su` by editing the file `/etc/pam.d/su` and un-commenting the line below. The file should look like this:

```

#%PAM-1.0
auth    sufficient  /lib/security/pam_rootok.so
# Uncomment following line to implicitly trust users in "wheel" group.
#auth    sufficient  /lib/security/pam_wheel.so trust use_uid
# Uncomment following line to require user to be in the "wheel" group.
auth    required    /lib/security/pam_wheel.so
auth    required    /lib/security/pam_stack.so service=system-auth
account required    /lib/security/pam_stack.so service=system-auth
password required    /lib/security/pam_stack.so service=system-auth
session required    /lib/security/pam_stack.so service=system-auth
session optional    /lib/security/pam_xauth.so

```

### Configure `/etc/fstab`

The file system table should be configured to limit the execution of `suid` files from certain file systems as well as the creation of device files on some of the file systems. RedHat Linux 7.1 uses a new format for the `fstab` table and includes the file system label instead of the the device file. The `fstab` file should look something like this:

```

LABEL=/          /          ext2  defaults          1 1

```

LABEL=/home	/home	ext2	nosuid,nodev	1 2
/dev/fd0	/mnt/floppy	auto	noauto,owner	0 0
LABEL=/tmp	/tmp	ext2	nosuid	
1 2				
LABEL=/var	/var	ext2	nosuid,nodev	1 2
LABEL=/var/spool/mail	/var/spool/mail	ext2	nosuid,nodev	
1 2				
none	/proc	proc	defaults	0 0
none	/dev/pts	devpts	gid=5,mode=620	0 0
/dev/hda6	swap	swap	defaults	0 0

## Remove unneeded programs and services

By default many programs and services are installed that are not needed. This includes utilities like apmd for power management, at which allows users to run commands at specific times, dhcpd a dhcp client daemon, and many more programs.

```

_____ /bin/rpm -e apmd (remove Power management utility)
_____ /bin/rpm -e at (remove User job queuing utility)
_____ /bin/rpm -e dhcpd (remove Dhcp client daemon)
_____ /bin/rpm -e ipchains (remove ipchains, iptables provides better firewalling)
_____ /bin/rpm -e gpm (remove general purpose mouse support)
_____ /bin/rpm -e lokkit (remove, generates ipchains only firewall scripts)
_____ /bin/rpm -e mouseconfig (remove mouse configuration utility)
_____ /bin/rpm -e pump (remove dhcp client)
_____ /sbin/chkconfig kudzu off (disable hardware auto detection utility)
_____ /sbin/chkconfig xinetd off (inetd daemon which nothing is using)

```

## Change permissions on programs

Many programs included by default do not need to be executed by users or setuid. To find all setuid programs use the following command:

```
_____ /usr/bin/find / -perm +4000 -print
```

Programs that need fixed permissions include:

```

_____ /bin/chmod 750 /sbin/depmod
_____ /bin/chmod 750 /sbin/inssmod
_____ /bin/chmod 750 /sbin/lsmmod
_____ /bin/chmod 750 /sbin/rmmmod
_____ /bin/chmod 750 /bin/setserial
_____ /bin/chmod 750 /usr/sbin/kbdconfig
_____ /bin/chmod 750 /usr/sbin/setup
_____ /bin/chmod 755 /usr/bin/ssh

```

```
_____ /bin/chmod 600 /etc/crontab
```

## Change /etc/issue

The RedHat 7.1 default /etc/issue and /etc/issue.net file give away a lot of information, including the RedHat release, and processor versions. Although we should not have any services running that use these files, except the console, they should be overwritten with less revealing information just to be safe. RedHat also regenerates this file every boot using a script in /etc/rc.d/rc.local so we must remove this script as well.

```
_____ /bin/echo 'Authorized users only\n'>/>/etc/issue  
_____ /bin/echo 'Authorized users only\n'>/>/etc/issue.net  
_____ /bin/echo >/etc/rc.d/rc.local
```

## Create an IPTables firewall

Linux kernel version 2.4.x includes new a new firewall method called IPTables or netfilter. This new version has support for connection tracking which is also known as stateful packet filtering, since it keeps track of each connection that has been established. We first need to load the Iptables rules using a firewall script. Create a script called /root/iptables.sh with the following contents, INET\_IP should be replaced with the machine Internet addresses. The script starts first by flushing all current rules, then specifically allows ports and services in. The default policy is to drop packets not defined in any of the accept rules. Some portions of the design influenced by Robert L. Ziegler's Web site: <http://linux-firewall-tools.com/ftp/firewall/rc.firewall.ipchains>. Many portions of this script were created with help from the netfilter mailing list, however I did not retain the email messages which aided in creating this script over a year ago.

```
#!/bin/sh  
# Firewall script  
/bin/echo Inserting firewall modules...  
/sbin/modprobe ip_tables  
/sbin/modprobe ip_conntrack  
/sbin/modprobe ip_conntrack_ftp  
/sbin/modprobe iptable_filter  
/sbin/modprobe ipt_limit  
/sbin/modprobe ipt_state  
  
/bin/echo Creating variables...  
# Variable Settings  
export ANY="0.0.0.0/0"      # Any system anywhere  
export INET="eth0"        # The Internet interface  
export INET_IP="10.0.0.2"  # The firewall's Internet address  
export ConnEst="-m state --state ESTABLISHED,RELATED"
```

```
export NotEst="-m state --state NEW,INVALID"
export Iptables=/sbin/iptables # Iptables program
```

```
/bin/echo resetting interface...
```

```
$Iptables -F INPUT      # flush existing INPUT rules
$Iptables -F OUTPUT     # flush existing OUTPUT rules
$Iptables -F FORWARD    # flush existing FORWARDing rules
```

```
/bin/echo securing interfaces...
```

```
# Prevent loopback attacks
$Iptables -A INPUT -j DROP -i $INET -s 127.0.0.0/8
$Iptables -A INPUT -j DROP -i $INET -d 127.0.0.0/8
$Iptables -A OUTPUT -j DROP -o $INET -s 127.0.0.0/8
$Iptables -A OUTPUT -j DROP -o $INET -d 127.0.0.0/8

# Prevent certain ping attacks
$Iptables -A INPUT -j DROP -i $INET -p icmp -s $ANY -d 255.255.255.255/32
$Iptables -A FORWARD -p icmp --icmp echo-request -m limit --limit 5/s -j
ACCEPT
```

```
/bin/echo setting allowed ports...
```

```
#----->Allow/Services<-----
# Allow ssh, smtp, to $INET_IP only
$Iptables -A INPUT -j ACCEPT -i $INET -p tcp -s $ANY -d $INET_IP --dport 22
$Iptables -A INPUT -j ACCEPT -i $INET -p tcp -s $ANY -d $INET_IP --dport smtp
# accept existing connections here
$Iptables -A INPUT -i $INET $ConnEst -j ACCEPT
# Drop all other packets from $inet_ip
$Iptables -A INPUT -i $INET -p UDP -j DROP
$Iptables -A INPUT -i $INET -j DROP
$Iptables -A FORWARD -i $INET -j DROP
/bin/echo operation complete !
```

Now we need to save these rules into the file that the RedHat 7.1  
/etc/rc.d/init.d/iptables script processes before configuring any network interfaces. This  
script will be run on every boot automatically.

```
_____ /sbin/iptables-save>/etc/sysconfig/iptables
_____ /sbin/chkconfig iptables on
```

There are some additional things we can do to further secure our tcp stack. This includes enabling syn cookies, ignoring icmp broadcasts, ignoring bogus icmp error messages, and not enabling tcp timestamps. We also specify to disable source routing for each valid interface and to log invalid packets. Edit /etc/rc.d/rc.local and put the following in the beginning of the file.



```

# enable kernel level protection
echo 1 > /proc/sys/net/ipv4/tcp_syncookies
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
echo 1 > /proc/sys/net/ipv4/icmp_ignore_bogus_error_responses
echo 0 > /proc/sys/net/ipv4/tcp_timestamps
# Turn on Source Address Verification
# from http://www.linuxdoc.org/HOWTO/IPCHAINS-HOWTO-5.html
for f in /proc/sys/net/ipv4/conf/*/rp_filter; do
    /bin/echo 1 > $f
done
for f in /proc/sys/net/ipv4/conf/*/log_martians; do
    /bin/echo 1 > $f
done

```

## Configuring Tripwire

Following all of your file configuration changes, Tripwire should be installed to notify you when any important files are changed. Your `/etc/tripwire/twpol.txt` should look like the `/etc/tripwire/twpol.txt` found under the appendix B. Create the `twpol.txt` file with the proper permissions and then edit.

```

_____ /bin/touch /etc/tripwire/twpol.txt
_____ /bin/chmod 600 /etc/tripwire/twpol.txt

```

Next run `/etc/tripwire/twinstall.sh`. Create a unique passphrase that is not the same as the root password or any other passwords you use. Tripwire will encrypt the policy file and database using this passphrase. For maximum security the Tripwire local passphrase should be different from the Tripwire site passphrase. Copy the text version of the policy file to removable media and remove the original file. Following this, initialize the Tripwire database and run `tripwire -check` to verify the Tripwire setup is correct. We also want to copy the tripwire local and site keys and the encrypted tripwire database to removable media.

```

_____ /etc/tripwire/twinstall.sh
_____ /bin/mount /dev/fd0 /mnt/floppy
_____ /bin/cp /etc/tripwire/twpol.txt /mnt/floppy
_____ /bin/rm /etc/tripwire/twpol.txt
_____ /usr/sbin/tripwire -init
_____ /usr/sbin/tripwire -check
_____ /bin/mkdir /var/lib/tripwire/tmp
_____ /bin/chmod -R 700 /var/lib/tripwire
_____ /bin/cp /etc/tripwire/sansbox-local.key /mnt/floppy
_____ /bin/cp /etc/tripwire/site.key /mnt/floppy
_____ /bin/cp /var/lib/tripwire/sansbox.twd /mnt/floppy
_____ /bin/umount /dev/fd0

```

Tripwire should run fairly often in order to detect modifications to files and promptly alert the administrator. Since we have a fairly powerful machine, we will set tripwire to run every 10 minutes and email reports to the administrator. We first create the appropriate permissions for the script.

```
_____/bin/touch /usr/local/sbin/tripwire-check  
_____/bin/chmod 700 /usr/local/sbin/tripwire-check
```

The following script will accomplish this and only mail violations and errors to the sansid account which monitors the Logcheck reports. Edit the script called /usr/local/sbin/tripwire-check and add the following:

```
#!/bin/sh  
umask 077  
#temporary directory  
trwtmp=/var/lib/tripwire/tmp  
/usr/sbin/tripwire --check &>${trwtmp}/tw.txt  
# mail tripwire report  
# We always mail the reports to root  
/bin/mail -s"Tripwire Report" root<${trwtmp}/tw.txt  
# Id we want errors and violations mailed to  
trwmail=sansid  
# messages to trigger violations or errors  
violations=`/bin/grep "No violations" ${trwtmp}/tw.txt`  
errors=`/bin/grep "No Errors" ${trwtmp}/tw.txt`  
#mail the reports if errors of violations found  
if [ -z "$violations" ]; then  
    /bin/mail -s"Tripwire Violations found" $trwmail<${trwtmp}/tw.txt  
elif [ -z "$errors" ]; then  
    /bin/mail -s"Tripwire Errors found" $trwmail<${trwtmp}/tw.txt  
fi
```

Next we add a cron entry to allow this script to run every approximately ten minutes. We want to make it slightly less obvious what intervals it runs at. Add the following line to /etc/crontab

```
_____ 2,11,21,33,42,51 * * * * root /usr/local/sbin/tripwire-check
```

## Perform a System Audit

The system should be audited to check for any potential security issues which may have arisen since the hardening process or any software security holes which may have appeared since you last patched. The system should be scanned using Nmap from an external source to verify only ports 22 for SSH and 25 for sendmail are open.

The system scan should be done from a secure isolated network. Nmap can be downloaded from <http://www.insecure.org/nmap> The scan should look something like this:

```
_____ /usr/bin/nmap -sTU -P0 -v sansbox
```

The results should look like the following (the IP address is fake):

```
Starting nmap V. 2.54BETA21 ( www.insecure.org/nmap/ )
Host sansbox (10.0.0.2) appears to be up ... good.
Initiating Connect() Scan against sansbox (10.0.0.2)
Adding TCP port 22 (state open).
Adding TCP port 25 (state open).
The Connect() Scan took 162 seconds to scan 1563 ports.
Initiating UDP Scan against sansbox (10.0.0.2)
The UDP Scan took 97 seconds to scan 1563 ports.
(no udp responses received -- assuming all ports filtered)
Interesting ports on sansbox (10.0.0.2):
(The 3124 ports scanned but not shown below are in state: filtered)
Port      State  Service
22/tcp    open   ssh
25/tcp    open   smtp
```

```
Nmap run completed -- 1 IP address (1 host up) scanned in 259 seconds
```

Nessus should also be utilized against the host to check for any potential software vulnerabilities or configuration errors. Nessus can be downloaded from <http://www.nessus.org/download.html> The procedure for using nessus will not be covered in this document.

We should test sendmail to verify relaying is not allowed. Use a mail client you are comfortable with for this test.

```
_____ Use a mail client such as pine http://www.washington.edu/pine/ or Netscape
http://www.netscape.com on another host and attempt to use sendmail as a
mail relay.
```

We need to verify that other hosts cannot connect to our ntp server, our firewall script should be blocking access. From a remote host type in:

```
_____ /usr/bin/ntpdate -s [ipaddress of sansbox]
_____ /bin/tail /var/log/messages (should see the message:
ntpdate[]: no server suitable for synchronization found
```

Next we test using SSH to verify that the pam\_tally configuration works. First make sure you are logged on and su to root. Then fail your login ten times, and verify that it works properly. Finally reset the user and verify that you can log in again. From a remote machine type in:

```
_____/usr/bin/ssh -l sansid sansbox (then type in the wrong password, repeat until 10 failed logins have occurred)
```

On sansbox type in:

```
_____/sbin/pam_tally --user sansid (you should see something like)
User sansid (703) has 10
```

On the remote host now try to log in with sansid and the valid password, it should fail.

```
_____/usr/bin/ssh -l sansid sansbox
```

On sansbox type in:

```
_____/sbin/pam_tally --user sansid --reset
```

On the remote host now try to log in with sansid and the valid password, it should work.

```
_____/usr/bin/ssh -l sansid sansbox
```

On sansbox, we also need to verify that users can only see their own processes, which will verify our kernel patch was successful. We should only see our own processes using the ps command. As a non-root user type in:

```
_____/bin/ps -aux
```

## Backup the System

A full system backup should be done after verifying that the system is functioning properly and that running nesses and nmap revealed no flaws. A replica of the system should be made using disk dumps, which will be tested on another system with identical hardware, to verify proper operation. The machine used to test for proper operation should not be networked, and should be completely sanitized after completing the procedure. Install the identical disks in the system. Use fdisk to make identical partition sizes for the replica disks. Bring the system to single user mode and begin the dumps:

```
_____/sbin/init 1
_____/bin/fdisk /dev/sdc
_____/bin/dd if=/dev/sda1 of=/dev/sdc1 bs=1k
_____/bin/dd if=/dev/sda6 of=/dev/sdc6 bs=1k
_____/bin/dd if=/dev/sda7 of=/dev/sdc7 bs=1k
```

```
_____ /bin/dd if=/dev/sda8 of=/dev/sdc8 bs=1k
_____ /sbin/fdisk /dev/sdd
_____ /bin/dd if=/dev/sdb1 of=/dev/sdd1 bs=1k
```

Now use the tape drive to perform a full backup of the system.

```
_____ /bin/mt -f /dev/st0 erase
_____ /bin/mt -f /dev/st0 rewind
_____ /bin/dump -0au -f /dev/st0 /
_____ /bin/dump -0au -f /dev/st0 /tmp
_____ /bin/dump -0au -f /dev/st0 /var
_____ /bin/dump -0au -f /dev/st0 /var/spool/mail
_____ /bin/dump -0au -f /dev/st0 /home
_____ /bin/mt -f /dev/st0 rewind
_____ /bin/mt -f /dev/st0 offline
```

The backup portion is complete, we now perform a filesystem check and verify the files. Following this we shut down the system, remove the replica disks and store them in a safe place.

```
_____ /sbin/fsck /dev/sdc1
_____ /sbin/fsck /dev/sdc5
_____ /sbin/fsck /dev/sdc6
_____ /sbin/fsck /dev/sdc7
_____ /sbin/fsck /dev/sdc8
_____ /sbin/fsck /dev/sdd1
_____ verify files on each partition
_____ /sbin/init 0
```

Last, we do a test restore from tape on another non-networked machine:

```
_____ /sbin/mke2fs /dev/sde1
_____ /bin/mkdir /mnt/diskrestore
_____ cd /mnt/diskrestore
_____ /bin/mt -f /dev/st0 rewind
_____ /sbin/restore -rv -f /dev/st0
_____ verify the files
_____ /bin/mt /dev/st0 rewoffl
```

Verify both the files on /mnt/diskrestore and the files on the disk dump drives. Erase the files from /mnt/diskrestore when finished verification. The tape and the disks to which the disk dumps were expanded to should be stored in a physically isolated safe locked area. Now that we have completed the installation we can now bring the system to the ISP co-location provider facility and plug it in to the network.

## Maintain the System

It is extremely important to have the most current security fixes for any operating system. Although all the hardening processes we have performed make it much more difficult for an attacker to compromise the system, a single security hole or bug in a piece of software can allow compromise. Several methods can be used to make sure you have the most up to date packages.

The RedHat watch list, notifies you when new packages are available:

\_\_\_\_\_ <http://www.redhat.com/mailling-lists/redhat-watch-list/index.html>

Security Focus Bugtraq, all security related alerts (select bugtraq checkbox):

\_\_\_\_\_ <http://www.securityfocus.com/about/feedback/subscribe.html>

RedHat Errata download site:

\_\_\_\_\_ <ftp://updates.redhat.com/7.1/en/os/i386>

Although Psionic logcheck does a great job at weeding out sometimes less critical data and alerting you to any potential issues, it is no substitute for reviewing your logs. System logs should be reviewed manually at least twice a day. The system logs that should be focused on are:

\_\_\_\_\_ /var/log/loginlog  
\_\_\_\_\_ /var/log/secure  
\_\_\_\_\_ /var/log/messages  
\_\_\_\_\_ /var/log/syslog  
\_\_\_\_\_ /var/log/kernel

The following may be reviewed on a less frequent basis:

\_\_\_\_\_ /var/log/cron  
\_\_\_\_\_ /var/log/maillog

© SANS Institute 2000 - 2005. Author retains full rights.

## Appendix A

/etc/syslog.conf after running Bastille hardening script:

```
# Log all kernel messages to the console.
# Logging much else clutters up the screen.
#kern.* /dev/console

# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;authpriv.none;cron.none /var/log/messages

# The authpriv file has restricted access.
authpriv.* /var/log/secure

# Log all the mail messages in one place.
mail.* /var/log/maillog

# Log cron stuff
cron.* /var/log/cron

# Everybody gets emergency messages, plus log them on another
# machine.
*.emerg *

# Save mail and news errors of level err and higher in a
# special file.
uucp,news.crit /var/log/spooler

# Save boot messages also to boot.log
local7.* /var/log/boot.log
##### BASTILLE ADDITIONS BELOW : #####
# Log warning and errors to the new file /var/log/syslog
*.warn;*.err /var/log/syslog

# Log all kernel messages to the new file /var/log/kernel
kern.* /var/log/kernel

# Log all logins to /var/log/loginlog
auth.*;user.*;daemon.none /var/log/loginlog

# Log additional data to the Alt-F7 and Alt-F8 screens (Pseudo TTY 7 and 8)

*.info;mail.none;authpriv.none /dev/tty7
authpriv.* /dev/tty7
*.warn;*.err /dev/tty7
kern.* /dev/tty7
mail.* /dev/tty8

*.* /dev/tty12
##### BASTILLE ADDITIONS CONCLUDED : #####
```

## Appendix B

/etc/tripwire/twpol.txt modified from default RedHat 7.1 installation:

```
# Global Variable Definitions # #
@@section GLOBAL
TWROOT=/usr/sbin;
TWBIN=/usr/sbin;
TWPOL="/etc/tripwire";
TWDB="/var/lib/tripwire";
TWSKEY="/etc/tripwire";
TWLKEY="/etc/tripwire";
TWREPORT="/var/lib/tripwire/report";
HOSTNAME=sansbox;
Email=tripwire;
@@section FS
SEC_CRIT      = $(IgnoreNone)-SHa ; # Critical files that cannot change
SEC_SUID      = $(IgnoreNone)-SHa ; # Binaries with the SUID or SGID flags set
SEC_BIN       = $(ReadOnly) ;      # Binaries that should not change
SEC_CONFIG    = $(Dynamic) ;      # Config files changed infrequently but accessed often
SEC_LOG       = $(Growing) ;      # Files that grow, but should never change ownership
SEC_INVARIANT = +tpug ;           # Dirs should never change permission or ownership
SIG_LOW       = 33 ;              # Non-critical files of minimal security impact
SIG_MED       = 66 ;              # Non-critical files of significant security impact
SIG_HI        = 100 ;             # Critical files of significant vulnerability

# Tripwire Binaries
(
  rulename = "Tripwire Binaries",
  severity = $(SIG_HI),
  emailto = $(Email)
)
{
  $(TWBIN)/siggen          -> $(SEC_BIN) ;
  $(TWBIN)/tripwire       -> $(SEC_BIN) ;
  $(TWBIN)/twadmin        -> $(SEC_BIN) ;
  $(TWBIN)/twprint        -> $(SEC_BIN) ;
}
(
  rulename = "Tripwire Data Files",
  severity = $(SIG_HI),
  emailto = $(Email)
)
{
  $(TWDB)                  -> $(SEC_CONFIG) -i ;
  $(TWPOL)/tw.pol          -> $(SEC_BIN) -i ;
  $(TWPOL)/tw.cfg          -> $(SEC_BIN) -i ;
  $(TWLKEY)/$(HOSTNAME)-local.key -> $(SEC_BIN) ;
  $(TWSKEY)/site.key       -> $(SEC_BIN) ;
  $(TWREPORT)              -> $(SEC_CONFIG) (recurse=0) ;
  ! $(TWDB)/tmp;           #ignore tripwire temporary directory
}
(
  rulename = "Invariant Directories",
  severity = $(SIG_MED),
  emailto = $(Email)
)
{
  /                          -> $(SEC_INVARIANT) (recurse = 0) ;
  /home                      -> $(SEC_INVARIANT) (recurse = 0) ;
  /etc                       -> $(SEC_INVARIANT) (recurse = 0) ;
}
```



```

}
# File System and Disk Administration Programs # #
(
  rulename = "File System and Disk Administration Programs",
  severity = $(SIG_HI),
  emailto = $(Email)
)
{
  /sbin/badblocks          -> $(SEC_CRIT) ;
  /sbin/e2fsck             -> $(SEC_CRIT) ;
  /sbin/debugfs           -> $(SEC_CRIT) ;
  /sbin/dumpe2fs          -> $(SEC_CRIT) ;
  /sbin/dump.static       -> $(SEC_CRIT) ;
  /sbin/e2label           -> $(SEC_CRIT) ;
  /sbin/fdisk             -> $(SEC_CRIT) ;
  /sbin/fsck              -> $(SEC_CRIT) ;
  /sbin/fsck.ext2         -> $(SEC_CRIT) ;
  /sbin/fsck.minix       -> $(SEC_CRIT) ;
  /sbin/hdparm            -> $(SEC_CRIT) ;
  /sbin/mkbootdisk        -> $(SEC_CRIT) ;
  /sbin/mke2fs            -> $(SEC_CRIT) ;
  /sbin/mkfs              -> $(SEC_CRIT) ;
  /sbin/mkfs.ext2        -> $(SEC_CRIT) ;
  /sbin/mkfs.minix       -> $(SEC_CRIT) ;
  /sbin/mkinitrd          -> $(SEC_CRIT) ;
  /sbin/mkswap            -> $(SEC_CRIT) ;
  /sbin/resize2fs        -> $(SEC_CRIT) ;
  /sbin/restore.static    -> $(SEC_CRIT) ;
  /sbin/sfdisk            -> $(SEC_CRIT) ;
  /sbin/tune2fs           -> $(SEC_CRIT) ;
  /sbin/update            -> $(SEC_CRIT) ;
  /bin/mount              -> $(SEC_CRIT) ;
  /bin/umount             -> $(SEC_CRIT) ;
  /bin/touch              -> $(SEC_CRIT) ;
  /bin/mkdir              -> $(SEC_CRIT) ;
  /bin/mknod              -> $(SEC_CRIT) ;
  /bin/mktemp             -> $(SEC_CRIT) ;
  /bin/rm                 -> $(SEC_CRIT) ;
  /bin/rmdir              -> $(SEC_CRIT) ;
  /bin/chgrp              -> $(SEC_CRIT) ;
  /bin/chmod              -> $(SEC_CRIT) ;
  /bin/chown              -> $(SEC_CRIT) ;
  /bin/cp                 -> $(SEC_CRIT) ;
  /bin/cpio               -> $(SEC_CRIT) ;
}
# Kernel Administration Programs # #
(
  rulename = "Kernel Administration Programs",
  severity = $(SIG_HI),
  emailto = $(Email)
)
{
  /sbin/depmod            -> $(SEC_CRIT) ;
  /sbin/ctrlaltdel        -> $(SEC_CRIT) ;
  /sbin/insmod            -> $(SEC_CRIT) ;
  /sbin/insmod.static     -> $(SEC_CRIT) ;
  /sbin/insmod_ksymlinks_clean -> $(SEC_CRIT) ;
  /sbin/klogd             -> $(SEC_CRIT) ;
  /sbin/ldconfig          -> $(SEC_CRIT) ;
  /sbin/minilogd          -> $(SEC_CRIT) ;
  /sbin/modinfo           -> $(SEC_CRIT) ;
  /sbin/sysctl            -> $(SEC_CRIT) ;
}
# Networking Programs # #

```

```

(
  rulename = "Networking Programs",
  severity = $(SIG_HI),
  emailto = $(Email)
)
{
  /sbin/arp                -> $(SEC_CRIT) ;
  /sbin/mingetty           -> $(SEC_CRIT) ;
  /sbin/ifconfig           -> $(SEC_CRIT) ;
  /sbin/ifdown             -> $(SEC_CRIT) ;
  /sbin/ifenslave          -> $(SEC_CRIT) ;
  /sbin/ifup               -> $(SEC_CRIT) ;
  /sbin/ipmaddr            -> $(SEC_CRIT) ;
  /sbin/iptables           -> $(SEC_CRIT) ;
  /sbin/iptunnel           -> $(SEC_CRIT) ;
  /sbin/netreport          -> $(SEC_CRIT) ;
  /sbin/plipconfig         -> $(SEC_CRIT) ;
  /sbin/ppp-watch          -> $(SEC_CRIT) ;
  /sbin/route              -> $(SEC_CRIT) ;
  /sbin/slattach           -> $(SEC_CRIT) ;
  /bin/ping                -> $(SEC_CRIT) ;
}
# System Administration Programs # #
(
  rulename = "System Administration Programs",
  severity = $(SIG_HI),
  emailto = $(Email)
)
{
  /sbin/chkconfig          -> $(SEC_CRIT) ;
  /sbin/fuser              -> $(SEC_CRIT) ;
  /sbin/halt               -> $(SEC_CRIT) ;
  /sbin/init               -> $(SEC_CRIT) ;
  /sbin/initlog            -> $(SEC_CRIT) ;
  /sbin/killall5           -> $(SEC_CRIT) ;
  /sbin/pwdb_chkpwd        -> $(SEC_CRIT) ;
  /sbin/rescuept           -> $(SEC_CRIT) ;
  /sbin/service           -> $(SEC_CRIT) ;
  /sbin/setsysfont         -> $(SEC_CRIT) ;
  /sbin/shutdown           -> $(SEC_CRIT) ;
  /sbin/sulogin            -> $(SEC_CRIT) ;
  /sbin/swapon             -> $(SEC_CRIT) ;
  /sbin/syslogd            -> $(SEC_CRIT) ;
  /sbin/unix_chkpwd        -> $(SEC_CRIT) ;
  /bin/pwd                 -> $(SEC_CRIT) ;
  /bin/uname               -> $(SEC_CRIT) ;
}
# Hardware and Device Control Programs # #
(
  rulename = "Hardware and Device Control Programs",
  severity = $(SIG_HI),
  emailto = $(Email)
)
{
  /sbin/hwclock            -> $(SEC_CRIT) ;
  /sbin/kbdrate            -> $(SEC_CRIT) ;
  /sbin/losetup             -> $(SEC_CRIT) ;
  /sbin/lspci              -> $(SEC_CRIT) ;
  /sbin/setpci             -> $(SEC_CRIT) ;
}
# System Information Programs # #
(
  rulename = "System Information Programs",
  severity = $(SIG_HI),

```

```

emailto = $(Email)
)
{
/sbin/consotype           -> $(SEC_CRIT) ;
/sbin/kernelversion       -> $(SEC_CRIT) ;
/sbin/runlevel            -> $(SEC_CRIT) ;
}
# Application Information Programs # #
(
  rulename = "Application Information Programs",
  severity = $(SIG_HI),
  emailto = $(Email)
)
{
/sbin/genksyms            -> $(SEC_CRIT) ;
/sbin/sln                 -> $(SEC_CRIT) ;
}
# Shell Related Programs # #
(
  rulename = "Shell Related Programs",
  severity = $(SIG_HI),
  emailto = $(Email)
)
{
/sbin/getkey              -> $(SEC_CRIT) ;
}
# OS Utilities # #
(
  rulename = "Operating System Utilities",
  severity = $(SIG_HI),
  emailto = $(Email)
)
{
/bin/cat                  -> $(SEC_CRIT) ;
/bin/date                 -> $(SEC_CRIT) ;
/bin/dd                   -> $(SEC_CRIT) ;
/bin/df                   -> $(SEC_CRIT) ;
/bin/echo                 -> $(SEC_CRIT) ;
/bin/egrep                -> $(SEC_CRIT) ;
/bin/false                -> $(SEC_CRIT) ;
/bin/fgrep                -> $(SEC_CRIT) ;
/bin/gawk                 -> $(SEC_CRIT) ;
/bin/gawk-3.0.6          -> $(SEC_CRIT) ;
/bin/grep                 -> $(SEC_CRIT) ;
/bin/true                 -> $(SEC_CRIT) ;
/bin/arch                 -> $(SEC_CRIT) ;
/bin/ash                  -> $(SEC_CRIT) ;
/bin/ash.static           -> $(SEC_CRIT) ;
/bin/basename             -> $(SEC_CRIT) ;
/bin/consolechars         -> $(SEC_CRIT) ;
/bin/dmesg                -> $(SEC_CRIT) ;
/bin/doexec               -> $(SEC_CRIT) ;
/bin/ed                   -> $(SEC_CRIT) ;
/bin/gunzip               -> $(SEC_CRIT) ;
/bin/gzip                 -> $(SEC_CRIT) ;
/bin/hostname             -> $(SEC_CRIT) ;
/bin/igawk                -> $(SEC_CRIT) ;
/bin/ipcalc               -> $(SEC_CRIT) ;
/bin/kill                 -> $(SEC_CRIT) ;
/bin/ln                   -> $(SEC_CRIT) ;
/bin/loadkeys             -> $(SEC_CRIT) ;
/bin/login                -> $(SEC_CRIT) ;
/bin/ls                   -> $(SEC_CRIT) ;
/bin/mail                 -> $(SEC_CRIT) ;
}

```

```

/bin/more          -> $(SEC_CRIT) ;
/bin/mv           -> $(SEC_CRIT) ;
/bin/netstat      -> $(SEC_CRIT) ;
/bin/nice         -> $(SEC_CRIT) ;
/bin/ps          -> $(SEC_CRIT) ;
/bin/rpm         -> $(SEC_CRIT) ;
/bin/sed         -> $(SEC_CRIT) ;
/bin/setserial   -> $(SEC_CRIT) ;
/bin/sleep       -> $(SEC_CRIT) ;
/bin/sort        -> $(SEC_CRIT) ;
/bin/stty        -> $(SEC_CRIT) ;
/bin/su          -> $(SEC_CRIT) ;
/bin/sync        -> $(SEC_CRIT) ;
/bin/tar         -> $(SEC_CRIT) ;
/bin/usleep      -> $(SEC_CRIT) ;
/bin/vi          -> $(SEC_CRIT) ;
/bin/zcat        -> $(SEC_CRIT) ;
}
# Critical Utility Sym-Links # #
(
  rulename = "Critical Utility Sym-Links",
  severity = $(SIG_HI),
  emailto = $(Email)
)
{
  /sbin/clock          -> $(SEC_CRIT) ;
  /sbin/kallsyms       -> $(SEC_CRIT) ;
  /sbin/ksyms          -> $(SEC_CRIT) ;
  /sbin/lsmmod         -> $(SEC_CRIT) ;
  /sbin/modprobe       -> $(SEC_CRIT) ;
  /sbin/pidof          -> $(SEC_CRIT) ;
  /sbin/swapoff        -> $(SEC_CRIT) ;
  /sbin/reboot         -> $(SEC_CRIT) ;
  /sbin/rmmmod         -> $(SEC_CRIT) ;
  /sbin/telinit        -> $(SEC_CRIT) ;
  /bin/awk              -> $(SEC_CRIT) ;
  /bin/dnsdomainname   -> $(SEC_CRIT) ;
  /bin/domainname      -> $(SEC_CRIT) ;
  /bin/ex              -> $(SEC_CRIT) ;
  /bin/gtar            -> $(SEC_CRIT) ;
  /bin/nisdomainname   -> $(SEC_CRIT) ;
  /bin/red              -> $(SEC_CRIT) ;
  /bin/rvi             -> $(SEC_CRIT) ;
  /bin/rview           -> $(SEC_CRIT) ;
  /bin/view            -> $(SEC_CRIT) ;
  /bin/ypdomainname    -> $(SEC_CRIT) ;
}
# Temporary directories # #
(
  rulename = "Temporary directories",
  recurse = false,
  severity = $(SIG_LOW),
  emailto = $(Email)
)
{
  /usr/tmp             -> $(SEC_INVARIANT) ;
  /var/tmp             -> $(SEC_INVARIANT) ;
  /tmp                 -> $(SEC_INVARIANT) ;
}
# Local files # #
(
  rulename = "User binaries",
  severity = $(SIG_MED),
  emailto = $(Email)
)

```

```

)
{
  /sbin                                -> $(SEC_BIN) (recurse = 1) ;
  /usr/local/bin                       -> $(SEC_BIN) (recurse = 1) ;
  /usr/local/sbin/                     -> $(SEC_BIN) (recurse = 1) ;
  /usr/sbin                             -> $(SEC_BIN) (recurse = 1) ;
  /usr/bin                              -> $(SEC_BIN) (recurse = 1) ;
}

(
  rulename = "Shell Binaries",
  severity = $(SIG_HI),
  emailto = $(Email)
)
{
  /bin/bsh                             -> $(SEC_BIN) ;
  /bin/csh                             -> $(SEC_BIN) ;
  /bin/sh                              -> $(SEC_BIN) ;
  /bin/bash                            -> $(SEC_BIN) ;
  /bin/tcsh                            -> $(SEC_BIN) ;
  /bin/bash2                           -> $(SEC_BIN) ;
}

(
  rulename = "Security Control",
  severity = $(SIG_HI),
  emailto = $(Email)
)
{
  /etc/group                           -> $(SEC_CRIT) ;
  /etc/security/                       -> $(SEC_CRIT) ;
}

(
  rulename = "Login Scripts",
  severity = $(SIG_HI),
  emailto = $(Email)
)
{
  /etc/csh.cshrc                       -> $(SEC_CONFIG) ;
  /etc/csh.login                       -> $(SEC_CONFIG) ;
  /etc/profile                          -> $(SEC_CONFIG) ;
}

# Libraries
(
  rulename = "Libraries",
  severity = $(SIG_MED),
  emailto = $(Email)
)
{
  !/usr/lib/perl5/man/whatis ;
  /usr/lib                              -> $(SEC_BIN) ;
  /usr/local/lib                        -> $(SEC_BIN) ;
}

# Critical System Boot Files          # #
(
  rulename = "Critical system boot files",
  severity = $(SIG_HI),
  emailto = $(Email)
)
{
  /boot                                 -> $(SEC_CRIT) ;
  /sbin/lilo                            -> $(SEC_CRIT) ;
}

```

```

    !/boot/System.map ;
    !/boot/module-info ;
}
# These files change every time the system boots ##
(
  rulename = "System boot changes",
  severity = $(SIG_HI),
  emailto = $(Email)
)
{
  !/var/run/ftp.pids-all ; # Comes and goes on reboot.
  /dev/log -> $(SEC_CONFIG) ;
  /dev/cua0 -> $(SEC_CONFIG) ;
  /dev/console -> $(SEC_CONFIG) -u ; #User ID may change
  /dev/tty2 -> $(SEC_CONFIG) ; # tty devices
  /dev/tty3 -> $(SEC_CONFIG) ; # are extremely
  /dev/tty4 -> $(SEC_CONFIG) ; # variable
  /dev/tty5 -> $(SEC_CONFIG) ;
  /dev/tty6 -> $(SEC_CONFIG) ;
  /dev/urandom -> $(SEC_CONFIG) ;
  /dev/initctl -> $(SEC_CONFIG) ;
  /var/lock/subsys -> $(SEC_CONFIG) ;
  /var/lock/subsys/random -> $(SEC_CONFIG) ;
  /var/lock/subsys/network -> $(SEC_CONFIG) ;
  /var/lock/subsys/anacron -> $(SEC_CONFIG) ;
  /var/lock/subsys/keytable -> $(SEC_CONFIG) ;
  /var/lock/subsys/netfs -> $(SEC_CONFIG) ;
  /var/lock/subsys/sshd -> $(SEC_CONFIG) ;
  /var/run -> $(SEC_CONFIG) ; # daemon PIDs
  /var/log -> $(SEC_CONFIG) ;
  /etc/issue.net -> $(SEC_CONFIG) -i ; # Inode changes
  /etc/ioctl.save -> $(SEC_CONFIG) ;
  /etc/issue -> $(SEC_CONFIG) ;
  /etc/.pwd.lock -> $(SEC_CONFIG) ;
  /etc/mtab -> $(SEC_CONFIG) -i ; # Inode changes
  /lib/modules -> $(SEC_CONFIG) ;
}
# These files change the behavior of the root account
(
  rulename = "Root config files",
  severity = 100
)
{
  /root -> $(SEC_CRIT) ; # Catch all additions
  /root/.bashrc -> $(SEC_CONFIG) ;
  /root/.bash_profile -> $(SEC_CONFIG) ;
  /root/.bash_logout -> $(SEC_CONFIG) ;
  /root/.bash_history -> $(SEC_CONFIG) ;
}
# Critical configuration files # #
(
  rulename = "Critical configuration files",
  severity = $(SIG_HI),
  emailto = $(Email)
)
{
  /etc/modules.conf -> $(SEC_BIN) ;
  /etc/crontab -> $(SEC_BIN) ;
  /etc/cron.hourly -> $(SEC_BIN) ;
  /etc/cron.daily -> $(SEC_BIN) ;
  /etc/cron.weekly -> $(SEC_BIN) ;
  /etc/cron.monthly -> $(SEC_BIN) ;
  /etc/default -> $(SEC_BIN) ;
  /etc/fstab -> $(SEC_BIN) ;
}

```

```

/etc/exports          -> $(SEC_BIN) ;
/etc/group-          -> $(SEC_BIN) ; # changes infrequent
/etc/host.conf       -> $(SEC_BIN) ;
/etc/hosts.allow     -> $(SEC_BIN) ;
/etc/hosts.deny      -> $(SEC_BIN) ;
/etc/protocols       -> $(SEC_BIN) ;
/etc/services        -> $(SEC_BIN) ;
/etc/rc.d/init.d     -> $(SEC_BIN) ;
/etc/rc.d            -> $(SEC_BIN) ;
/etc/mail.rc         -> $(SEC_BIN) ;
/etc/motd            -> $(SEC_BIN) ;
/etc/passwd          -> $(SEC_CONFIG) ;
/etc/passwd-         -> $(SEC_CONFIG) ;
/etc/profile.d       -> $(SEC_BIN) ;
/etc/rpc             -> $(SEC_BIN) ;
/etc/sysconfig       -> $(SEC_BIN) ;
/etc/nsswitch.conf   -> $(SEC_BIN) ;
/etc/hosts           -> $(SEC_CONFIG) ;
/etc/xinetd.conf     -> $(SEC_CONFIG) ;
/etc/xinetd.d        -> $(SEC_CONFIG) ;
/etc/inittab         -> $(SEC_CONFIG) ;
/etc/resolv.conf     -> $(SEC_CONFIG) ;
/etc/syslog.conf     -> $(SEC_CONFIG) ;
/etc/ssh/sshd_config -> $(SEC_CONFIG) ;
/etc/ssh             -> $(SEC_CONFIG) ;
}
# Critical devices # #
(
  rulename = "Critical devices",
  severity = $(SIG_HI),
  emailto = $(Email),
  recurse = false
)
{
  /dev/kmem          -> $(Device) ;
  /dev/mem           -> $(Device) ;
  /dev/null          -> $(Device) ;
  /dev/zero          -> $(Device) ;
  /proc/devices     -> $(Device) ;
  /proc/net         -> $(Device) ;
  /proc/sys         -> $(Device) ;
  /proc/cpuinfo     -> $(Device) ;
  /proc/modules     -> $(Device) ;
  /proc/mounts     -> $(Device) ;
  /proc/dma         -> $(Device) ;
  /proc/filesystems -> $(Device) ;
  /proc/pci         -> $(Device) ;
  /proc/interrupts -> $(Device) ;
  /proc/ioports     -> $(Device) ;
  /proc/kcore       -> $(Device) ;
  /proc/self        -> $(Device) ;
  /proc/kmsg        -> $(Device) ;
  /proc/stat        -> $(Device) ;
  /proc/ksyms       -> $(Device) ;
  /proc/loadavg     -> $(Device) ;
  /proc/uptime      -> $(Device) ;
  /proc/locks       -> $(Device) ;
  /proc/version     -> $(Device) ;
  /proc/meminfo     -> $(Device) ;
  /proc/cmdline     -> $(Device) ;
  /proc/misc        -> $(Device) ;
}
# Rest of critical system binaries
(

```

```
rulename = "OS executables and libraries",
severity = $(SIG_HI),
emailto = $(Email)
)
{
    /bin          -> $(SEC_BIN) ;
    /lib          -> $(SEC_BIN) ;
}
=====
#
# Copyright 2000 Tripwire, Inc. Tripwire is a registered trademark of
# Tripwire,
# Inc. in the United States and other countries. All rights reserved.
#
# Linux is a registered trademark of Linus Torvalds.
#
# UNIX is a registered trademark of The Open Group.
#
=====
#
# Permission is granted to make and distribute verbatim copies of this
# document
# provided the copyright notice and this permission notice are preserved on
# all
# copies.
#
# Permission is granted to copy and distribute modified versions of this
# document under the conditions for verbatim copying, provided that the entire
# resulting derived work is distributed under the terms of a permission notice
# identical to this one.
#
# Permission is granted to copy and distribute translations of this document
# into another language, under the above conditions for modified versions,
# except that this permission notice may be stated in a translation approved
# by
# Tripwire, Inc.
#
# DCM
```

© SANS Institute 2000 - 2005. Author retains full rights.



## References

---

Securing Linux Step By Step Version 1.0; The Sans Institute, guided and edited by Lee E. Brozman, Allied Technology Group, Inc. and David A. Ranch, Trinity Designs

Russel, Rusty. "Linux 2.4 Packet Filtering HOWTO". Version 1.19. May 26 2001. URL: <http://netfilter.filewatcher.org/unreliable-guides/packet-filtering-HOWTO/index.html>

Russel, Rusty. "Linux IPCHAINS-HOWTO". Version 1.0.8, July 4 2000. URL: <http://www.linuxdoc.org/HOWTO/IPCHAINS-HOWTO.html>

Spender. "Getrewted General Security Guide". URL: <http://www.getrewted.net/Documents/general.txt>

RedHat Inc. "RedHat Linux 7.1, The Official Red Hat Linux Reference Guide". URL: <http://www.redhat.com/support/manuals/RHL-7.1-Manual/ref-guide/>

Beale, Jay. "Using Linux 2.4 Firewalling – Building a Firewall with Netfilter". February 19 2001. URL: <http://www.securityportal.com/articles/netfilter20010219.html>

Beale, Jay. "Bastille Linux: A Walkthrough". June 6 2000. URL: <http://www.securityfocus.com/frames/?focus=linux&content=/focus/linux/articles/linux-bastille.html>

Gray, Michael. "Build a Secure Web Server Using Red Hat Linux Version 6.2 Step By Step". URL: [http://www.sans.org/y2k/practical/Michael\\_Gray\\_GCUX.doc](http://www.sans.org/y2k/practical/Michael_Gray_GCUX.doc)

Ward, Brian. "The Linux Kernel HOWTO". Version 3.0. July 15 2001. URL: <http://www.linuxdoc.org/HOWTO/Kernel-HOWTO.html>

Ziegler, Robert L. "Linux LAN and Firewall FAQ, Configuring an Internet Firewall and Home LAN with Linux". September 28, 1999. URL: <http://linux-firewall-tools.com/linux/faq/index.html>

# Upcoming Training

Click Here to  
**{Get CERTIFIED!}**



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