Interested in learning more about security?

SANS Institute
InfoSec Reading Room

This paper is from the SANS Institute Reading Room site. Reposting is not permitted without express written permission.

How-To Make Linux System Auditing a Little Easier

Auditing your system and network covers an important aspect of security, detection. It is your last line of defense. It is crucial that you have in place a means of determining the state of your system and to detect unauthorized logins and system changes. To accomplish this there are several programs and utilities that are made available but using them all on a daily basis and over a period of time can be an over whelming task if you don't design a good strategy. The design must be simple enough, yet effective, so that...

Copyright SANS Institute
Author Retains Full Rights

AD

GIAC Security Essentials Certification (GSEC)
Practical Assignment Version 1.4

How-To Make Linux System Auditing a Little Easier

Paul J. Santos

9/15/02

© SANS Institute 2002, Author retains full rights.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

Contents

Abstract 3
Introduction 3
Applications 4
Logcheck (logging) 4
Tripwire 4
chkrootkit 5
GnuPG 5
sysstat 6
md5sum 6
System Utilities 7
ps 7
ifconfig 7
find 8
top 8
ls 8
netstat 8
uptime 8
last 8
who 8
df 9
Putting it Together 9
Prep Work 9
Daily Run 12
Maintenance 12
Conclusion 12
Appendix 13
sysaudit.sh 13
hcreate.sh 18
References 19

© SANS Institute 2002, Author retains full rights

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

Abstract

Auditing your system and network covers an important aspect of security, detection. It is your last line of defense. It is crucial that you have in place a means of determining the state of your system and to detect unauthorized logins and system changes.
To accomplish this there are several programs and utilities that are made available but using them all on a daily basis and over a period of time can be an over whelming task if you don't design a good strategy. The design must be simple enough, yet effective, so that it can be used regularly and over a long period of time.

In this paper I will talk about the various programs and utilities that can be used to audit your Linux system and how to put them all together in one script to make daily system auditing a little easier.

Introduction

After setting up and hardening your Linux system the next logical step before putting it into production is to create a baseline of data that represents its current state. Auditing is one method to achieve this. Through the auditing process, you learn about your system using various system utilities and programs and then record this information in a secure manner to use as a security detection system and for use in future investigations and forensics. Thus, it becomes very important to design an audit routine that becomes very easy to run daily, is easily archived and secured, and is effective in detecting critical unauthorized changes to your system.
The most obvious way to achieve this is to implement an all-in-one solution that meets all these goals. Unfortunately, I have yet to see one although there are many that come close but they always fail to be complete. Most likely because we all have special system implementations and it becomes harder to find a good solution all in one package.
My approach is to use the various readily available utilities and programs and gather them together into one simple routine that is not only effective but can also be customized to particular system uses. My solution is to implement various auditing programs, and system utilities and call on them from one simple script on a daily basis and have the resulting audit log securely mailed to a central management station where it can be reviewed and archived.
This concept is not new and there have been many implementations of it. This paper simply details my approach and one that I have implemented and have been using.
I call my implementation script "sysaudit.sh" and it contains simple system commands, calls on applications to run, and then securely emails me the results on a daily basis.
I have included a step-by-step setup process and sample sysaudit.sh script at the end of this paper and will briefly describe each application and command in the following sections.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

Applications

The following are a review of the applications that are used in system auditing, what they achieve, and how I implement them in my solution (script).

Logcheck (logging)

One of the most common and easily implemented methods of auditing is monitoring log files. Checking your system log files regularly is the first step in creating a good auditing scheme.
For the most part, Linux logging is typically all setup for you and all you need is a routine that you can follow to monitor them regularly. Another important consideration is central logging. If you have multiple systems, you should configure each syslog facility to send a copy of its logs to a central dedicated log server or management station. This will make the analysis easier and backup of the logs files for future use.
Most distributions now have Logcheck installed to help you monitor your system logs. Log tools are very important because they help manage the process by filtering information so the log files don't become useless information just taking up space and time.
Logcheck, now called LogSentry, automatically parses your log files and mails you any security violations that it finds. Through the use of configuration files, you tell Logcheck what to look for and what to ignore in the logs.

Logcheck can be configured to run regularly from Cron but in my implementation I call on Logcheck from the sysaudit.sh script. Once launched, Logcheck performs its functions then emails the results to the predefined email address. For email addresses, I typically configure all applications and scripts to send to the root address and then simply configure root's .forward file to forward its mail to the central management station.

Logcheck can be downloaded from http://www.psionic.com/products/ or www.rpmfind.net.

Tripwire

Taken from http://sourceforge.net/projects/tripwire, "Tripwire is a policy driven file system integrity checking tool that allows system administrators to verify the integrity of their data".
Tripwire is a great tool to keep track of your file system state although; its report can be a bit verbose at times. As you will see when implementing the sysaudit.sh script, you will not always have to review the Tripwire report in detail and mainly use it only when an unauthorized change is suspected. Properly archived, Tripwire reports can also help create a good audit trail.
Once invoked from the sysaudit.sh script, Tripwire will run its integrity check and then append its report to the sysaudit.sh log.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

Open source version of Tripwire can be downloaded from http://www.tripwire.org/ or www.rpmfind.net.

chkrootkit

The Chkrootkit program is another free system integrity checking tool that was designed specifically to look for signs of installed root kits. Root kits are commonly used by hackers to create back doors, and perform other hacking activity, to systems that they have somehow gained access to.
Any signs of root kit detection should immediately raise flags and alert you of system comprise, at which time, steps should be taken to secure the system and its logs and begin detail analysis of current and archived system logs.
Chkrootkit is a very simple program and outputs a very simple report. Invoked from the sysaudit.sh script, Chkrootkit runs its system check and then has its report appended to the sysaudit.sh log. This will be one the critical sections that should be looked at in the emailed logs in a daily basis. Any detection here will then warrant a complete analysis of the entire sysaudit.sh log and possibly a number of archived logs as well.

Chkrootkit can be downloaded from http://www.chkrootkit.org/.

GnuPG

GnuPG is the free open source version of the popular PGP software, developed by Philip Zimmermann, and is an RFC2440 compliant application. It uses public key cryptography to encrypt and sign mail messages for secure email exchange.

GnuPG is not so much used here as an audit tool but a way to secure the mailed logs themselves. This is especially useful in cases where the logs are being mailed to a management station outside of the home network or in the DMZ. The audit itself contains a tremendous amount of information that would be very useful to a would be attacker.

In the sysaudit.sh script, the near completed audit log is first encrypted to the public key of the administrator before it is mailed out. This protects all information in the mailed log itself incase it is somehow sniffed out of the network. The administrator then decrypts the mailed log using his GnuPG private key and then proceeds to review it and finally archive it.

GnuPG can be downloaded from http://www.gnupg.org/ or www.rpmfind.net.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

sysstat

The sysstat program, developed by Sebastien Godard, is a system accounting program used to collect statistical data on various system components including CPU, Network, Disk, and process related activities.
The concept here is detection through monitoring of system performance. This utility also has the added ability to provide trend analysis for use in capacity planning.
The main Sysstat command that I use here is the sar command, which is used to create a report on collected data specifying various different counters. The two counters that I will use in the audit process are the processor utilization and network activity. Through the use of these two counters we can detect suspicious activity and the times that they occur.
In the sysaudit.sh script, the sar command is used to create a report on processor and network activity and is appended to the log. This information can be used to either detect suspicious activity or to help analyze a suspected intrusion.
Please note that use of Sysstat may degrade system performance and could clutter up the sysaudit.sh log file a bit although, later you will find that the log does not always have to be reviewed in detail. It also acts as a good method to archive system data.

Sysstat can be downloaded from http://perso.wanadoo.fr/sebastien.godard/ or www.rpmfind.net.

md5sum

The md5sum command performs file integrity operations on files by creating one-way hashes (message-digest) for each file. These hashes can then be used to run checks on the installed files on the system. Md5sum has been very popular in the use on verifying various types of downloaded files and programs.

This integrity-checking program is the foundation of which the sysaudit.sh script that I created is based on. The md5sum program is used here to create integrity checks on various system key files including all the above programs and their configuration files that they use. The idea is that it is much simpler to secure just a single integrity database instead of various others like Tripwire and the like. As long as the md5sum database file checksums pass then all the other programs can be trusted but, if any checks fail with md5sum then it immediately flags that there has been an intrusion and that complete auditing and analysis is required. At this point, any auditing applications in the system cannot be trusted up to the point of the failed md5sum checksum.
This also adds to the concept that the less immune files you have on the system the easier to detect intrusion it becomes. In this solution, the only program and files that are immune, via read-only media, is the md5sums and the sysaudit.sh script. All other audit programs are installed and setup in their default locations

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

making it much simpler and quicker to setup the audit system. The md5sum will keep tabs on these files and confirm their integrity.

The process of creating an md5sum checksum file is performed by putting commands in a simple script that will compute md5sum hashes on various system files and then output them to a single file. This file is then secured and used to run future md5sum checksums comparing them to the current system. In the sysaudit.sh script, md5sum command is used with –check argument to create a report of all passed checksums and any failed ones. This is then appended to the sysaudit.sh log. This section of the audit log will be the top priority section to be checked on a daily basis. Any discrepancies here will immediately flag a problem.

These and other setup processes will be detailed later in this paper.

Md5sum is part of the textutils package and can be downloaded from www.rpmfind.net. Note that md5sum has been part of most Linux standard installations and you will most likely already have it installed.

System Utilities

Common system utilities are the most readily available tools, since they are part of the base system, and provide an enormous amount of information on the current system state.
They include ps, ifconfig, find, top, ls, netstat, uptime, last, who, and df. These tools together help to create valuable system audit report.
These utilities are part of most Linux systems and all have man pages that you can refer to for their detail usage. They will only be described briefly here. Please see their respective man page for further reading. You can also query http://www.sonic.net/cgi-bin/man for online man pages.

ps

The process status utility reports on various system processes that are running. One of the most important activities when getting to know and securing your system is knowing what processes (programs and services) are running on your system. Comparing a ps output from a previous baseline, any new running process detected should flag a problem.

ifconfig

The ifconfig (interface configuration) command is used to either change or query current network parameters. The main things to check here is correct IP address and mask, any excessive transmission errors, and if the interface was put in promiscuous mode (PROMISC).

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

find

The find command is an invaluable tool for searching through the file system for various different types of files. Find can be used to find files based on expressions and various file attributes such as permissions and modification time.
In the sysaudit.sh script find is used to look for any unauthorized SUID or GUID files, accounts with no passwords, un-owned files, world-writable files and directories, and core files.

top

The top utility displays information on processor and memory usages. It will show which processes are running and how much CPU time and memory they are using. This tool can be used to look for any unauthorized process using up too much memory and CPU time. Any process that is using excessive resources, that normally do not, should indicate a problem.

ls

The ls command is used to provide a list of directory contents and their permissions. In the sysaudit.sh script, ls displays permissions of certain critical system files, like /etc/passwd, and is used to confirm that these permission have not be altered. Note that /etc/passwd should be readable only by the root account.

netstat

The netstat command is used to display current network connections and ports that are listening for connections. The main task here is to confirm that you don't have any new or unauthorized services listening for incoming connections. It is also used to keep track on services that are authorized to be running and what ports they are configured for.

uptime

The uptime command is used to report how long the system has been running without a reboot. It also gives you information on system load averages. The uptime output will indicate if the system has been shutdown or rebooted, which would indicate either a hardware issue or unauthorized access.

last

The last command reports the list of users who have logged on recently. If you are not using the root account for administration or local access then root listed in this report would indicate a possible system breach.

who

The who command reports users that are currently logged on. Again, this is used to look for any unauthorized access.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

df

The df command (disk free) is used to keep track of hard drive space usage on currently mounted file systems. Any file systems close to capacity would indicate an issue, authorized or not.

Putting it Together

As mentioned earlier, one of the concepts of making auditing easier to use and setup is to have one program (md5sum) to keep integrity checks on the others. In my implementation, I used a floppy disk, which is mounted as read-only (ro) to hold the md5sum and script files. Other options include CD media or a dedicated partition on the hard drive that can be mounted as read-only.
The other concept, which is the basis of this paper, is to combine all the tools that you have to perform a system audit into one effective routine using a simple script.
The process is as follows. After creating and setting up the script the system is setup, via cron, to run the script every morning (or night pending on your need, preference, or log rotation time). The script runs and goes through all of its commands and then creates a final report (log) and then mails it out securely (using GnuPG) to the central administration station. Administrator receives this report, and others if there are more systems, and reviews it. Pending on the time that he has or day of the week that it is, he may just briefly look over only sections that he may be concerned with but, always including at minimum, the md5sum checksums. Any discrepancies in the md5sum checksum should alert the administrator of a possible problem and indicate that the logs need to be reviewed in detail.
After reviewing the report the administrator archives it in his machine, and possibly another central log server, for future reference and analysis.

The following describes the basic process of setting up the auditing applications and sysaudit.sh script and includes a sample script (see appendix), which you are free to use. Please fine-tune and test the script on non-production system first!
The setup process is broken up into three sections consisting of prep work, daily run, and maintenance. Please note that the Linux system should have already been hardened and has not been in production yet. Be sure to check if you don't already have the following applications installed. Use "rpm –q <rpm>" command to see if it is already installed.
The following was performed on a redhat 7.0 Intel system as root user.

Prep Work

Prepare a floppy disk by performing low level and high level format:

1. fdformat /dev/fd0
2. mkfs -c -t ext2 /dev/fd0
3. mount /mnt/floppy

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

Set up the directory structure on the floppy. Note that this elaborate structure of hidden directories and files are only to add a layer (a small one) of security. Create the flowing directories on the floppy:

mkdir –p /mnt/floppy/.sysa/.hcheck

Copy the md5sum binary, sysaudit.sh, and hcreate.sh scripts to the floppy disk following the same directory structure used in the script:

1. cp sysaudit.sh /mnt/floppy/.sysa/.sysaudit.sh
2. cp /usr/bin/md5sum /mnt/floppy/.sysa/md5sum
3. cp hcreate.sh /mnt/floppy/.sysa/.hcheck/.hcreate.sh

Edit /etc/fstab to mount the floppy as read-only (ro). Should look like this:

/dev/fd0 /mnt/floppy auto nouser,ro 0 0

Be sure to set the BIOS not to boot to floppy first incase of power fail and reboot.

Install and configure Tripwire (www.tripwire.org or www.rpmfind.net):

1. rpm –ivh tripwire.rpm (installs it if you don't already have it).
2. /etc/tripwire/twinstall.sh (runs the setup script. Enter pass phrases when prompted).
3. tripwire –init (creates your initial integrity database).
4. See tripwire docs on making adjustments on the configuration and profile files and updating the database.

Install and configure Logcheck (or substitute with your favorite log parser).

1. Install Logcheck from tarball or RPM (see www.psionic.com or www.rpmfind.net).
2. Adjust the /usr/bin/logcheck.sh as needed (e.g. email address to send log to).
3. Optionally remove the cron job that was added for Logcheck, since we will be launching it from the sysaudit.sh script.
4. Adjust Logcheck's igore and violations files as needed. See Logcheck docs for details on these.

Download and install chkrootkit (http://www.chkrootkit.org).

1. tar –xzf chkrootkit.tar.gz (extracts the tar ball).
2. make sense (compiles it).
3. ./chkrootkit (test run).
4. Note that for the purpose of the md5sum script (hcreate.sh), I installed chkrootkit in the /root directory.

Install and configure GnuPG. See www.rpmfind.net for the rpm and www.gnupg.org for docs. The following assumes that GnuPG is already installed.

1. gpg –gen-key (generates your key pair if not already done).

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

2. gpg –import <admin's key file.asc> (import the administrator's public key into GnuPG key ring. This is the admin key to which the audit log will be encrypted and sent to).
3. gpg –edit-key <admin's imported key name/email address> (this opens to key edit mode which you then type "sign" so as to trust the admin key).
4. See GnuPG docs for details.

Install and configure sysstat. It can be downloaded from http://perso.wanadoo.fr/sebastien.godard/ or www.rpmfind.net.

1. Rpm –ivh <sysstat.rpm> (installs sysstat rpm).
2. Verify that cron jobs and /var/log/sa directory was properly setup.
3. See http://perso.wanadoo.fr/sebastien.godard/ for details.

Adjust the sysaudit.sh script as necessary (e.g. file paths, email address, and GnuPG key to use).

Setup the cron job to run the sysaudit.sh script daily. Note that the "2>" is used to redirect the error output to a temp file, which is then later appended to the log. The following will run every day at 6:05 AM:

1. crontab –e (enters the crontab editor).
2. Enter: 05 06 * * * /mnt/floppy/.sysa/.sysaudit.sh 2>/root/.errortmp
3. :wq (writes and exits VI editor).

Adjust the hcreate.sh (short for hash create) script as necessary and run it to create the initial md5um file. Note that hcreate.sh script is made hidden with "." and that the floppy needs to be temporarily mounted as read-write (rw) in order to create the md5sum file.

1. mount -o remount,rw /mnt/floppy (remounts floppy as rw).
2. /mnt/floppy/.sysa/.hcheck/.hcreate.sh (runs the .hcreate.sh script).
3. mount -o remount,ro /mnt/floppy (remounts the floppy as ro).

Set restrictive permissions on critical files. Note that this will also remove SUID on some binaries. Do not perform these commands if you need other regular users to use them.

1. chmod –R 700 /mnt/floppy/.sysa
2. chmod 700 /bin/mount /bin/umount /usr/bin/crontab
3. chmod –R 700 /etc/cron.d
4. chmod 700 /etc/cron.daily
5. chmod 600 /etc/crontab

To add another layer of security you can also write-protect the floppy disk and restrict floppy access to root by editing the /etc/security/console.perms file. See the console.perms man page for details.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

Daily Run

The daily run consists of the sysaudit.sh script being run from cron, or manually launched for testing, and the administrator reviewing the log from his management station. There are basically two ways you can view the daily log. First, you should review the log in detail at least once or twice a week. Second, you can view only the most important sections pending on the time that you have. As noted earlier, during this time the most important sections to review are the md5sum, chkrootkit, and tripwire outputs. This will tell you if any unauthorized changes have been made on any files or binaries in the system.
All logs are then archived in the remote admin station for use of future reference and analysis.

Maintenance

Maintenance involves updating the databases after authorized changes have been made and verified to the system. This involves updating the tripwire database and then updating the md5sum hash file. The process is as follows:

Update the Tripwire database. Note that if you get any errors trying to run the update simply run the "tripwire –check" manually to create an updated report. Reports are kept in /var/lib/tripwire/report directory.

1. tripwire –update –r <last report.twr> (runs the tripwire update mode)
2. :wq (writes and quits VI).
3. Enter pass phrase when prompted to complete the update.

You must then update the md5sum hash file on the floppy. Be sure that last log report does not show any unauthorized or suspicious system changes.

1. mount -o remount,rw /mnt/floppy
2. /mnt/floppy/.sysa/.hcheck/.hcreate.sh
3. mount -o remount,ro /mnt/floppy

Conclusion

As you can see, combining all available utilities and applications into one routine, which is implemented here as a simple script, can make auditing a Linux system not only easier in a daily basis but also over a long period of time.
Please note that for security completeness you should also periodically perform security assessments on you Linux systems, which include port scans and vulnerability scans.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

Appendix

The following are my example scripts that I use. You are free to use them as you see fit. Please fine-tune and test them on non-production systems first!

The sysaudit.sh is the script that is run from cron and has all the commands that makes up the audit.
The hcreate.sh (hash create) script is used create the md5sum hash file.
Adjust as necessary.

sysaudit.sh

```
#!/bin/bash

# After reading this script you'll get the idea/concept. You can simply
# add/remove any commands that you wish to run.
# Script simply runs each command and appends the output of that
# command to the log file. Log file is mailed out at the end.

# I created all the temp files in /root but you could also put them in
# a sticky bit directory for extra security. You can also customize all of the file
# names and directory structure to create a unique setup.

# The following was successfully run on redhat 7.0 but was also easily modified to
# run on redhat 7.3 and FreeBSD 4.x.

# variables:

TMPLOG="/root/.sysaudit.log"
SERVERNME=`/bin/hostname`
ERRLOG="/root/.errortmp"
CHKTMP="/root/.chktmp"
TMPLOGE="/root/.sysaudit.log.asc"
ENCRPKEY="<admins email address>"

# Mail page header:

echo "##############################################" >$TMPLOG
echo "                                              " >>$TMPLOG
echo "                System Audit                  " >>$TMPLOG
echo "                                              " >>$TMPLOG
echo "##############################################" >>$TMPLOG

# log the time and date
/bin/date >> $TMPLOG
# system info:
/bin/uname -a >> $TMPLOG
# System name:
echo "    Server:  " >>$TMPLOG
echo $SERVERNME >>$TMPLOG

# Functions

# This function is called to insert a line/cmd seperator in the log file.
```


© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

```
# makes it easier to read the emailed log file.

seperator()
{

echo "                          "  >>$TMPLOG
echo " *************************" >>$TMPLOG
echo "                          "  >>$TMPLOG

}

# Commands to run:

################ File System ################

seperator
echo "File system check:" >>$TMPLOG
echo number of SUID files: >>$TMPLOG
/usr/bin/find / -type f \( -perm -04000 -o -perm -02000 \) -ls |wc -l >>$TMPLOG

seperator
echo world writable files: >>$TMPLOG
/usr/bin/find / -type f \( -perm -2 \) -ls |wc -l >>$TMPLOG

seperator
echo world writable directories: >>$TMPLOG
/usr/bin/find / -type d \( -perm -2 \) -ls |wc -l >>$TMPLOG

seperator
echo un-owned files: >>$TMPLOG
/usr/bin/find / -nouser -o -nogroup >>$TMPLOG

seperator
echo number of core files: >>$TMPLOG
/usr/bin/find / -name core | wc -l >>$TMPLOG

seperator
echo files modified in the last day: >>$TMPLOG
/usr/bin/find /etc -mtime -1 >>$TMPLOG

seperator
echo Hard drive usage: >>$TMPLOG
/bin/df -h >>$TMPLOG

seperator
echo "permissions on /etc/shadow and others:" >>$TMPLOG
/bin/ls -l /etc/shadow >>$TMPLOG
/bin/ls -l /usr/bin/crontab >>$TMPLOG
/bin/ls -l /bin/mount >>$TMPLOG
/bin/ls -l /bin/umount >>$TMPLOG
/bin/ls -l /etc/crontab >>$TMPLOG
/bin/ls -ld /etc/cron.daily >>$TMPLOG
/bin/ls -ld /etc/cron.d >>$TMPLOG

seperator
# I ran rpm verification only on critical network tools since
```


© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

```
# running a complete rpm database check can be verbose.
echo "rpm check on net-tools (null = OK):" >>$TMPLOG
/bin/rpm -V net-tools >>$TMPLOG

seperator
echo Script files info: >>$TMPLOG
/bin/ls -alhR /mnt/floppy/.sysa >>$TMPLOG

############# System Processes #############

seperator
echo "                System Processes     " >>$TMPLOG
echo Processes running: >>$TMPLOG
/bin/ps aux >>$TMPLOG

echo number of running processes: >>$TMPLOG
/bin/ps aux | wc -l >>$TMPLOG

seperator
echo services set to run: >>$TMPLOG
 /sbin/chkconfig --list >>$TMPLOG

seperator
echo modules loaded: >>$TMPLOG
/sbin/lsmod >>$TMPLOG

seperator
echo TOP >>$TMPLOG
/usr/bin/top -b -n 1 >>$TMPLOG

################ Networking ############

seperator
echo "               Networking         " >>$TMPLOG
echo IFCONFIG: >>$TMPLOG
/sbin/ifconfig >>$TMPLOG

seperator
echo NETSTAT current connections: >>$TMPLOG
/bin/netstat -an >>$TMPLOG

seperator
echo "RPC Info (null= no RPC services running= ok):" >>$TMPLOG
/usr/sbin/rpcinfo -p >>$TMPLOG

seperator
# I run a ping to check status on critical servers.
# Adjust or comment out.
echo Ping >>$TMPLOG
/bin/ping -c 2 <insert IP here> >>$TMPLOG

################ System Accounting #######

seperator
echo "              System Accounting    " >>$TMPLOG
echo system uptime: >>$TMPLOG
```


© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

```
/usr/bin/uptime >>$TMPLOG

seperator
echo last log on list: >>$TMPLOG
/usr/bin/last >>$TMPLOG

seperator
echo who is logged on now: >>$TMPLOG
/usr/bin/who >>$TMPLOG

seperator
echo cron jobs: >>$TMPLOG
/usr/bin/crontab -l >>$TMPLOG

seperator
echo UID 0 accounts: >>$TMPLOG
/bin/awk -F: '($3 == 0) { print $1 }' /etc/passwd >>$TMPLOG

seperator
echo "system accounting/trend analysis:" >>$TMPLOG
 /usr/bin/sar -u -n DEV >>$TMPLOG

################## Applications #############

seperator
# Note here that logcheck once invoked has no std output so its report
# is mailed seperately. I don't know about logcheck but, logwatch has
# the ability to configure a std output that you could append to this log.
# Substitute with your favorite log parser.
echo "                Applications     " >>$TMPLOG
echo "Logcheck (mailed seperately):" >>$TMPLOG
 /usr/bin/logcheck.sh

seperator
echo chkrootkit says: >>$TMPLOG
/root/chkrootkit >>$TMPLOG

# md5sum
# I used a temp file so I could grep out keywords (FAILED) since
# md5sum --check is very verbose. This will tell you how many total hashes,
# how many were OK, and which ones failed.
# Please note that you will see /etc/mtab file fail because with this
# implementation, the md5sum file was created when floppy was rw.

seperator
echo md5sum hashes reports: >>$TMPLOG
/mnt/floppy/.sysa/md5sum --check /mnt/floppy/.sysa/.hcheck/.hashdb >$CHKTMP 2>>$TMPLOG
echo "number of files checked OK:" >>$TMPLOG
/bin/grep -c OK $CHKTMP >>$TMPLOG
echo "number of files failed checksum:" >>$TMPLOG
/bin/grep FAILED $CHKTMP >>$TMPLOG

# Adding Tripwire stuff
seperator
echo Tripwire report: >>$TMPLOG
```


© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

```
 /usr/sbin/tripwire --check >>$TMPLOG


################  LOGS  ###############

# Add any other system log files that you wish to
# review and archive.

seperator
echo "                System Logs     " >>$TMPLOG
echo  the MESSAGES log: >>$TMPLOG
/bin/cat /var/log/messages >>$TMPLOG

# append the sysaudit.sh error log at the end
seperator
echo Error log >>$TMPLOG
/bin/cat $ERRLOG >>$TMPLOG

################# Done ################

# At then end of the Audit the log file is mailed out.

# encrypt the log file first:

 /usr/bin/gpg --batch -ea -r $ENCRPKEY $TMPLOG

# Then mail it. I typically send it to root then configure root's .forward file.
# Adjust as necessary.

 /bin/mail -s sysaudit root <$TMPLOGE

# Clean up

 /bin/rm -f $ERRLOG
 /bin/rm -f $TMPLOG
 /bin/rm -f $TMPLOGE
 /bin/rm -f $CHKTMP
```

© SANS Institute 2002, Author retains full rights.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

hcreate.sh

```
#!/bin/bash

# hcreate.sh script
# Creates md5sum checksums on files and puts them in an ascii file (.hashdb).
# This file is then put on read-only (ro) media.
# Note the md5sum does not perform recursively so, ignore the subdirectories errors.
#
# Adjust as necessary.

# Define variables

HASHCMD="/mnt/floppy/.sysa/md5sum"
DBFILE="/mnt/floppy/.sysa/.hcheck/.hashdb"

$HASHCMD /bin/* > $DBFILE
$HASHCMD /sbin/* >> $DBFILE
$HASHCMD /etc/* >> $DBFILE
$HASHCMD /etc/tripwire/* >> $DBFILE
$HASHCMD /var/lib/tripwire/* >> $DBFILE
$HASHCMD /usr/sbin/tripwire >> $DBFILE
$HASHCMD /root/chkrootkit >> $DBFILE
$HASHCMD /var/lib/rpm/* >> $DBFILE
$HASHCMD /etc/logcheck/* >> $DBFILE
$HASHCMD /usr/sbin/* >> $DBFILE
$HASHCMD /usr/bin/* >> $DBFILE
$HASHCMD /usr/bin/logcheck.sh >> $DBFILE
$HASHCMD /etc/rpm/* >> $DBFILE
$HASHCMD /etc/ssh/* >> $DBFILE
$HASHCMD /etc/httpd/conf/* >> $DBFILE
$HASHCMD /mnt/floppy/.sysa/* >> $DBFILE
$HASHCMD /mnt/floppy/.sysa/.hcheck/.hcreate.sh >> $DBFILE

# set perm to rw only by root and group root.
chmod 640 $DBFILE
```


© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

References

Whelan, Paul. "*Linux Security Auditing.*" 01 June 2001. URL: http://rr.sans.org/audit/linux_sec.php (14 Sept. 2002)

Hutchison, Bill. "Adding Chkrootkit to Your Unix Auditing Arsenal". 26 Feb 2001. URL: http://rr.sans.org/malicious/chkrootkit.php (08 Sept. 2002).

Ashley, Mike. "The GNU Privacy Handbook". 1999. URL: http://www.gnupg.org/gph/en/manual.html (14 Sept. 2002).

Wen, Bobby. "Open-Source Intrusion-Detection Tools for Linux." 01 Oct 2000. URL: http://www.linuxjournal.com/article.php?sid=4054 (08 Sept. 2002).

Howard, Eric. "Filesystem auditing for your Linux box using md5sum." URL: http://www.geocities.com/eric_howard_linux_site/md5sum.htm (08 Sept. 2002).

Cooper, Mendel. "Advanced Bash-Scripting Guide." 13 July 2002. URL: http://www.tldp.org/LDP/abs/html/ (31 Aug. 2002).

Mourani, Gerhard. "Securing and Optimizing Linux." 07 June 2000. URL: http://www.linuxsecurity.com/docs/PDF/Securing-Optimizing-Linux-RH-Edition-1_3.pdf (5 Aug. 2002).

Naidu, Krishni. "Auditing Linux". URL: http://www.sans.org/SCORE/checklists/AuditingLinux.doc (08 Sept. 2002).

Cole, Eric. *Hackers Beware*. First Ed. Indianapolis: New Riders, August 2001.

Hatch, Brian. Lee, James. Kurtz, George. *Hacking Linux Exposed*: *Linux Security Secrets & Solutions*. Berkeley: Osborne/McGraw-Hill, 2001.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

Last Updated: May 21st, 2013

Upcoming SANS Training

Click Here for a full list of all Upcoming SANS Events by Location

SANS Malaysia @ MCMC 2013	Cyberjaya, MY	Jun 03, 2013 - Jun 08, 2013	Live Event
SANS Pen Test Berlin 2013	Berlin, DE	Jun 03, 2013 - Jun 08, 2013	Live Event
Industial Control Systems Security Training - Houston	Houston, TXUS	Jun 10, 2013 - Jun 15, 2013	Live Event
Security Impact of IPv6 Summit 2013	Washington, DCUS	Jun 14, 2013 - Jun 16, 2013	Live Event
SANSFIRE 2013	Washington, DCUS	Jun 14, 2013 - Jun 22, 2013	Live Event
SANS Canberra 2013	Canberra, AU	Jul 01, 2013 - Jul 13, 2013	Live Event
Digital Forensics & Incident Response Summit 2013	Austin, TXUS	Jul 09, 2013 - Jul 16, 2013	Live Event
SANS London Summer 2013	London, GB	Jul 09, 2013 - Jul 16, 2013	Live Event
SANS Rocky Mountain 2013	Denver, COUS	Jul 14, 2013 - Jul 20, 2013	Live Event
SANS Mumbai 2013	Mumbai, IN	Jul 22, 2013 - Jul 27, 2013	Live Event
SEC528 SANS Training Program for the CompTIA® New Advanced Security Practitioner (CASP) Certification	Washington, DCUS	Jul 22, 2013 - Jul 26, 2013	Live Event
SANS San Francisco 2013	San Francisco, CAUS	Jul 29, 2013 - Aug 03, 2013	Live Event
SANS SEC 560: Network Penetration Testing @ Bangalore 2013	Bangalore, IN	Aug 05, 2013 - Aug 10, 2013	Live Event
SANS Boston 2013	Boston, MAUS	Aug 05, 2013 - Aug 10, 2013	Live Event
Critical Security Controls Summit	Washington, DCUS	Aug 12, 2013 - Aug 18, 2013	Live Event
Industrial Control Systems Security Training - DC	Washington, DCUS	Aug 12, 2013 - Aug 16, 2013	Live Event
SANS Thailand 2013	Bangkok, TH	Aug 19, 2013 - Aug 31, 2013	Live Event
SANS Virginia Beach 2013	Virginia Beach, VAUS	Aug 19, 2013 - Aug 30, 2013	Live Event
Mobile Device Security Summit 2013	OnlineCAUS	May 30, 2013 - Jun 06, 2013	Live Event
SANS OnDemand	Books & MP3s OnlyUS	Anytime	Self Paced