# Global Information Assurance Certification Paper

# GIAC Certified Unix Security Administrator

## Practical Assignment
## Version 1.9 Option 1.0
## Securing Unix Step by Step

# Developing a Secure and Portable Snort Sensor based on Red Hat 9




**Submitted by:**

**Rick Larabee**
**June 20, 2003**

## Abstract

The purpose of the paper is to look into what goes into building a secure and portable snort sensor. It will look at the software and hardware that the system will be built upon, walk through the step-by-step procedure of securely installing and configuring the software and operating system, what maintenances needs to happen to the system after it is deployed, and finally test cases to check the configuration and security of this system.

## Description of System

The system built for this paper will serve as a network monitoring station that can be deployed easily for monitoring internal subnets and/or specific users. In many networks, IDS systems are deployed at choke points or near high value assets to watch the majority of the traffic. There are cases where a network analyst may need to monitor a specific user or user subnet and that connection may either be local or on a remote subnet. The system will be a network monitoring station that uses Red Hat Linux version 9 as its operating system and Snort[1] as the software to capture and monitor the data. Since this system will be designed to be deployable and automatic, there is no need for user accounts on the system other than the generic Administrator account. The overall structure of the network monitoring system will consist of this station and a Management Station. The management station will provide a centralized point for managing this sensor and any others that are deployed. The Management Station can also be one that is already in place to support permanent IDS sensors. The process of building the management station will not be in the scope of this paper but will be referenced in key spots where applicable.

### Software

**Tripwire**[2] is a file integrity monitor. It uses different hashing routines and attribute checking to verify whether files have changed or if any files have been added or deleted from a directory.

**OpenSSH**[3] was developed to provide a secure replacement to the remote administration utilities for Unix, such as telnet, ftp, and the r-commands. OpenSSH encrypts all of the traffic between two systems, while the other utilities are in clear text.

**IPTables**[4] is a stateful packet filtering software used on the 2.4.x and 2.5.x linux kernels. It will function as a host-based firewall.

**NTP**[5] stands for Network Time Protocol. NTP will be utilized to keep the system's clock synchronized with the other monitoring systems for event correlation.

[1] Snort Home Page, URL: http://www.snort.org
[2] Tripwire Home Page, URL: http://www.tripwire.com
[3] OPENSSH Home Page, URL: http://www.openssh.org
[4] Netfilter Home Page, URL: http://www.netfilter.org/
[5] NTP Home Page, URL: http://www.ntp.org/

**Port Knocking**. Martin Krzywinski developed a client and daemon in perl so that port knocking could be used. Port Knocking is a method of using closed ports to open a port. By attempting to connect to a series of closed ports, the log monitoring daemon will decrypt the series of closed ports and if the series authenticates properly, the daemon can add a rule to either open or close a port by adding rules to the host level firewall, in this case iptables. Once the port is open the client can use the service listening on that port.[6]

### Hardware

The Optiplex GX150 is a small form factor PC that is manufactured by Dell. It is approximately 12.5"x13.5"x3.5" making it easy to be moved or shipped around. The system will only contain one network card, both for maintenance and for monitoring, in order to keep the deployment simple and something that anyone can accomplish with only one page of instructions.

| | |
|---|---|
| Processor | Intel PIII 1 gigahertz |
| RAM | 512 megabytes |
| Hard Drive | 20 gigabytes IDE |
| Network card | 10/100 3com 3c905x |

## Risk analysis of the system

This appliance will only be connected to the internal network inside the perimeter firewall. If this box was going to be used to monitor traffic outside of the firewall, a second NIC card would be installed to meet best practice procedures. Best practice would include one NIC configured for the internal network used to manage the box and another NIC, not assigned an ip address, used to monitor to external traffic.

Because the system is going to be configured to monitor the internal network and will probably sit either in a data center or in a network closet somewhere, the highest risk to this system would be buffer overflows in the daemons that are network accessible. After buffer overflows, trojaned software is next, followed by monitoring of the management network traffic, running out of drive space, and finally physical access.

### Buffer Overflows

Buffer overflows have been thoroughly discussed in numerous articles, discussion groups, and white papers. While buffer overflows are not the topic of this paper, a short review is in order so the reader may understand the risk

[6] Krzywinski, Martin. Port Knocking home site http://mkweb.bcgsc.bc.ca/portknocking/

involved. When a program has allocated a certain amount of memory, say 256 kb, for a variable and a user submits 512 kb of data to the variable, the buffer (memory) overflows, possibly overwriting other code on the stack. In the data that overflows, malicious code can be inserted that the operating system will run and may give access to the system with whatever privileges the program has. Frequently this will be root level privileges, giving full access to the operating system and anything else on that system. This is a very basic description of buffer overflows. Links that give a better, more detailed description of a buffer overflow can be found in the appendix.

The two services that are of concern for this system are the Snort and ssh daemon. Buffer overflows are defeated by proper programming that accomplished input validation. Normally, incorrect programming is found and corrected by newer releases to the program. Therefore, the best method to prevent buffer overflows is to install the latest version of the software for each daemon and establish a proactive patching process. Another method to help mitigate the exposure to buffer overflows, is to set up IPTables to deny any and all incoming traffic unless it has been initiated from the local machine or by using Port Knocking. Since IPTables is not effective in protecting the Snort process, Snort will be running under a different user other than root, by using chroot. Chroot stands for "change root", which basically states change the default root directory of a file system, typically "/", to something further down the tree, like "/var/snort". The idea is to prevent an exploited service from being able to access any of the system's critical processes. Initially snort has to be started as root so that it can put the network card it promiscuous mode, but will change to another user "snort" and to a lower level root file system "/var/snort" If the process is exploited, it should only be able to do limited damage.

In coordination with IPTables, port knocking will be utilized as the first access control to the ssh daemon. Port knocking is using a combination of closed ports to access a daemon. In this case, iptables will be setup to drop all network traffic that is destined for the machine except for return traffic that was initiated by this machine. The port knocking client will then attempt to connect to a combination of 8 ports between 745-1000. The daemon watches for the denied packets showing up in a log file. The daemon will decrypt the series of denied packets and verify the checksum to make sure it is a legitimate request. Legitimate requests will allow the submitted ip address access to the port submitted by the client. Even if someone was able to guess the correct Blowfish[7] password and submit a correct series of closed ports to open a port, tcpwrappers (2nd level of access control) is configured to only allow the management station's ip address to access the ssh daemon. Finally, ssh will be configured to only allow public key authentication.

[7] The Blowfish Encryption Algorithm, Counterpane Labs, URL: www.counterpane.com/blowfish.html

### Trojaned software

A trend that has happened recently is altering of source code on distribution servers. When a user downloads and installs the altered software, additional code inserted will provide some kind of backdoor for the cracker to be able to come back in. Recent cases of this have been verified with tcpdump, libpcap[8], and sendmail[9]. To prevent trojaned software, verification of the code will be accomplished either by using md5 hashes or pgp keys. It is also important to check the posted md5 sums from multiple sources, in case the same person that modified the code also changed the web page. Pgp public keys and signatures are considered to be an even more secure method of verifying the code.

### Network Monitoring

If someone has installed some type of network monitoring utility, they could potentially see the management traffic going in between the management station and the snort sensor. The port knocking traffic is encrypted with the blowfish encryption. The actual management of the snort sensor is done with ssh, so all traffic is encrypted. Also, any file transfers are done using scp, a utility of the openssh suite. SCP stands for secure copy and allows encrypted file transfer between two systems.

### Running out of drive space

Running out of drive space is a low priority because Snort should not consume a lot of drive space as long as it is configured and managed correctly. If Snort is being used in intrusion detection mode with a properly configured rule set, the amount of traffic captured should be minimal. If it is being used to capture specific traffic the captured traffic could be significant. In this case, Snort can be restarted to start a new log file and old log files can be archived on another system.

### Physical Access

Physical access is always a problem with any system. This system should be deployed in a network closet or data center whenever possible. It will be either connected to a spanned port on a switch, a hub with other users, or will be connected to a user's network jack that is then connected to a hub where the user's system and the snort sensor will be connected to the same hub.

---

[8] Latest libpcap & tcpdump sources from tcpdump.org contain a trojan., Houston's Linux User Group, http://www.hlug.org/trojan/

[9] Houston's Linux User Group, Latest libpcap & tcpdump sources from tcpdump.org contain a trojan.
http://www.hlug.org/trojan/

A bios password will not be setup on this machine because when this system is sent out, it will just have a power cord, network connection and turned on. There will be no way to enter a password. The vulnerabilities with this setup are the system could be unplugged, either the network connection or power, someone could steal the system or components out of the system, or someone could hook up a keyboard to the system to try to break into it. Hopefully, the network closet or data center will only have a select group of individuals that have access to it thereby mitigating just the average user being able to damage anything. The idea behind this system is something that can be easily setup in the local headquarters or be shipped out to a remote location for someone to setup for the Administrator. Although the risk of having someone unplugging the system from the power outlet is something that cannot be mitigated, it can be monitored. If someone unplugged the system, plugged it back in and restarted it, there would be new log entries in the log files that could have associated alerts. In addition, the method used to communicate with the sensor will provide clues that the system has been restarted. To prevent someone from stealing the system or components out of the system, a kensington cable lock and pad lock will be used. The pad lock will be hooked to the systems chassis lock and the cable lock will be wrapped around the network rack. To mitigate someone from hooking a keyboard up to the system or having access to the floppy drive or cdrom, a metal plate is placed over the ports on the back of the machine except for the power and network jack. Once the machine is configured the floppy and cdrom drives are disconnected from the motherboard to prevent bypassing the internal Linux security.

## Step by step guide

Note: The setup process for the Snort Sensor will be accomplished by using two separate systems. One system, the Staging Server, will be used to download programs, verify them using md5 hashes and pgp keys, and compile the code. The staging server serves two purposes. One is to prevent the sensor from having to be on the network before it is fully configured and patched. The other is that the sensor, once in production, will not have any development tools on it to limit the abilities of an attacker should the system be compromised. For example, an attacker would be unable to download source code to the system and compile it.

The staging server will have a CD burner installed, will have Red Hat Linux 9 installed as its OS, and will have development tools installed as required for the installation. This paper assumes the reader is familiar with Linux system commands, knows how to burn a CD, and how to use both the vi and pico editor to modify or create files.

### Media download and integrity test

The first step to building the sensor is downloading Red Hat Linux 9 from Red Hat's ftp site ftp://ftp.redhat.com/, a mirror site ftp://rpmfind.net, or using the new peer to peer app BitTorrent. The files that are needed are:

ftp://rpmfind.net/linux/redhat/9/en/iso/i386/shrike-i386-disc1.iso
ftp://rpmfind.net/linux/redhat/9/en/iso/i386/shrike-i386-disc2.iso
ftp://rpmfind.net/linux/redhat/9/en/iso/i386/shrike-i386-disc3.iso

These files will be downloaded, verified, and burned to a cdrom on the staging server.

Verify that the integrity of the iso files by getting the MD5Sums for them from ftp://ftp.redhat.com/pub/redhat/linux/9/en/iso/i386/MD5SUM
For the three files, the MD5 hashes are:

```
400c7fb292c73b793fb722532abd09ad  shrike-i386-disc1.iso
6b8ba42f56b397d536826c78c9679c0a  shrike-i386-disc2.iso
af38ac4316ba20df2dec5f990913396d  shrike-i386-disc3.iso
```

Once the files are downloaded, to verify the md5 hash, at a command prompt in the directory that the files are located type each of the following commands, retrieving the md5sum from each:

```
md5sum shrike-i386-disc1.iso
md5sum shrike-i386-disc2.iso
md5sum shrike-i386-disc3.iso
```

The MD5SUM file has been signed by Red Hat's pgp public key, which can be obtained from http://www.redhat.com/security/db42a60e.txt to verify the contents of the file.

The validity of the MD5SUM file can be verified using the pgp key. On the staging server, download the key from Red Hat's site, import the key into the gpg database, and verify the file as shown below.

```
wget http://www.redhat.com/security/db42a60e.txt
gpg --import db42a60e.txt
gpg --verify MD5SUM
```

The line that says where the signature is good or not is:

```
Signature made Tue 01 Apr 2003 03:07:31 PM CST using DSA key ID
DB42A60E
gpg: Good signature from "Red Hat, Inc <security@redhat.com>"
```

It is extremely important that the Snort Sensor not be attached to the network until it is completely configured and patched. Any updates or compiled software will be moved via floppy disk or CD from the staging server to the sensor to prevent any vulnerability from being exploited during the setup phase.

Once the CDs are prepared, the Snort Sensor bios should be setup to boot from the cdrom drive first. Then, insert disc1 in the Snort Sensor and restart the system.

When the initial boot screen comes up, it gives you the option of:

1. Install or upgrade Red Hat in graphical mode
2. Install or upgrade Red Hat in text mode
3. Or choose from the function keys options access to specific boot options.

Select Option 1: Install Red Hat in graphical mode by either hitting enter or doing nothing and waiting for the default timeout.

It will then ask to test the CD media or skip. Test all 3 CDs by selecting "OK" here. The test looks at the embedded MD5Sum file on the CD to verify that the CD was burned correctly. After it finishes with the first CD, it will either give a grade of PASS or FAILED. At that point it allows for testing of the other 2 CDs

After all media has been tested, re-insert disc1 and select "Continue".

**Installation**

When the installation program starts, select the options as outlined below:

Welcome Screen
Click Next here.

Keyboard Configuration
Select U.S. English (Default), and click Next.

Language Selection
Select "English (English)", which is the default select and click Next.

Mouse Configuration
Select "No mouse". This system will never have a mouse attached to it except for during setup, click Next to move on.

Installation Type
Select "Custom" and click Next.

Disk Partitioning Setup
Select Manually partition with Disk Druid and click Next

Disk Setup

If there are any partitions listed make sure to delete them before starting the install. To delete the partition, click on the partition so that it is highlighted and then click the delete button.

Once all of the partitions are deleted, it should look like:

```
Hard Drives
      /dev/had
            Free        Free Space        19093 1     2434
```

Click the New button, the windows that is presented is the “Add Partition” screen. For this screen on the first one

Mount Point: “/boot”
File System Type: “ext3” – the default
Allowable Drives: there is only one drive so it is automatically checked
Size (MB): “75” – Note: Red Hat recommends having the boot partition at no less than 75 megabytes
Fixed Size is checked
Check for bad blocks is checked
Click OK

Back at the Disk Setup screen, click New again

Mount Point: “/”
File System Type: “ext3” – the default
Allowable Drives: do not change
Size (MB): 1000
Fixed Size is checked
Check for bad blocks is checked
Click OK

Click New again

Mount Point: “/tmp”
File System Type: “ext3” – the default
Allowable Drives: do not change
Size (MB): 500
Fixed Size is checked
Check for bad blocks is checked
Click OK

Click New again

Mount Point: (leave empty)
File System Type: “swap”
Allowable Drives: do not change

Size (MB): 1024 – double the amount of ram.
Fixed Size is checked
Check for bad blocks is checked
Click OK

Click New again

Mount Point: "/var"
File System Type: "ext3" – the default
Allowable Drives: do not change
Size (MB): 100 - default
Fill to maximum allowable size is checked
Check for bad blocks is checked
Click OK

Once finished it should look like:

```
Hard Drives
      /dev/had
            /dev/hda1   /boot ext3  (checkmark) 78    1     10
            /dev/hda2          swap  (checkmark) 1028  11    141
            /dev/hda3   /tmp  ext3  (checkmark) 502   142   205
            /dev/hda4          Extended          17485 206   2434
            /dev/hda5   /     ext3  (checkmark) 502   206   269
            /dev/hda6   /var  ext3  (checkmark) 16983 269   2434
```

Click Next

The file systems are partitioned out to prevent a denial of service to the operating system. For instance, the /var directory will hold all of the log files and network capture files. If it was not a separate partition from the root file system "/" and a malicious user was able to fill the log files to consume all of the hard drive space, the system may become unusable. With /var being a separate partition, the only thing that would stop is any new events being logged.

Boot Loader Configuration

Leave the default selection of
"Red Hat Linux" /dev/hda5 selected

Place a checkmark beside "Use a boot loader password". When checked, type in the select password for Grub. The reason for doing this is the case someone does gain access to the console and tries to reboot the system; they will not be able to send different options to the kernel during boot up.
Click Next

Network Configuration

The system when deployed will obtain an IP address from whatever network it is attached to. This is to prevent additional maintenance overhead. If this sensor is deployed to a non-dhcp enabled segment, the box will be configured with an IP address before it is sent out. Under Hostname click Manually and enter “SnortSensor”
Click Next

Firewall Configuration
Select No firewall
The host firewall will be setup at a later time.

Additional Language Support
Select English(USA) and click next

Time Zone Selection
Select whatever time zone this sensor is located in. For this case, “America/Chicago Central Time” is selected.
Click Next

Set Root Password
Select a root password for this sensor, remember to use a complex password. Best practice is to use one that is at least 9 characters in length, with an upper case, lower case, number, and special character included. For instance M0r3$4m3% is a good password that can be easily remembered (More money for me), but would not be easily guessed or brute forced.
Click Next

Authentication Configuration
Leave as the default – Check both:
Enable MD5 passwords – MD5 passwords allows longer passwords to be used, up to 256 characters.
Enable Shadow passwords – separates the actually password from the password file to allow only root to read access the shadow file (contains the encrypted password). This prevents a normal user from reading the /etc/passwd and being able to run a password cracking utility on the encrypted passwords. /etc/passwd has to be able to be read by all users.

Under NIS, LDAP, Kerberos 5, and SMB nothing should be checked
Click Next

Package Group Selection
Uncheck everything that is check by default, the only boxes that should be checked are “Minimal – Choose this group to get the minimal possible set

of packages. Useful for creating small router/firewall boxes, for example" and "Select individual packages" at the bottom of the window.
The total install size should read 474M
Click Next

Individual Package Selection

Click Flat View, the only packages that need to be selected are:

APMD
CRONTABS
DEVLABEL
DHCLIENT
DIFFUTILS
GNUPG
HESIOD
IPTABLES
LIBCAP
LIBPCAP
LOGROTATE
LOGWATCH
LSOF
M4
MAILCAP
MAILX
NTP
OPENSSH
OPENSSH-CLIENTS
OPENSSH-SERVER
PERL
PERL-FILTER – needed by Perl
PERL-TIME-HiRes
PROCMAIL
QUOTA
SENDMAIL
SENDMAIL-CF
SLOCATE
TCPWRAPPERS
TMPWATCH
TRIPWIRE
UNZIP
UTEMPTER
VIXIE-CRON
ZIP

Total Install size should be 414M
Click Next.

The reason for selecting RPM packages and not compiling the source code is for ease of management.

Installing Packages

The status of the install will be displayed while the kernel and software packages are installed.

Boot Diskette Creation

Select the default – Yes I would like to create a boot diskette
Click Next
Insert a blank diskette in the floppy drive and click on Make Boot disk

Congratulations

The CD should eject and click Exit.

The system will reboot, at the boot loader prompt, either hit <enter> and just wait for it to boot.

**Update all packages**

On the staging server, download all of the latest updates for Red Hat 9 by using the command:

```
ncftp ftp.rpmfind.net
```

Download all of the updates from the i386, i686, and noarch directories using the commands:

```
cd linux/redhat/updates/9/en/os/i386
mget *

cd linux/redhat/updates/9/en/os/i686
mget *

cd linux/redhat/updates/9/en/os/noarch
mget *
```

At the writing of this document, there were approximately 79 updated packages. Some of these packages are different versions, such as the kernel with bigmem support, smp, or for single processor systems. Other packages were compiled for both the i386 and i686 architecture. The packages compiled for the i686 architecture should be used as they have been optimized for the newer processors. Therefore, the files that can be deleted are:

```
glibc-2.3.2-27.9.i386.rpm
kernel-2.4.20-13.9.i386.rpm
kernel-2.4.20-13.9.i686.rpm
kernel-2.4.20-18.9.i386.rpm
```

```
kernel-BOOT-2.4.20-13.9.i386.rpm
kernel-doc-2.4.20-13.9.i386.rpm
kernel-smp-2.4.20-13.9.i686.rpm
openssl-0.9.7a-5.i386.rpm
```

After all of the rpms have been downloaded, verify all of the packages by using the following command:

```
rpm --checksig *.rpm
```

Red Hat's pgp key should already be installed from when the MD5SUM files for the install cds were checked.

Copy Red Hat's pgp key over to the snort sensor with a floppy disk and import it using this command:

```
gpg --import /mnt/floppy/db42a60e.txt
```

Burn all of the rpms to a cd. Mount the cd that contains the updated rpms in the snort sensor and execute:

```
cd /mnt/cdrom
rpm -Fvh *.rpm
```

The "F" switch for the rpm command will "freshen" the packages that are currently installed only if there is a newer version or better optimized package (i386 vs i686).

**File Cleanup and package cleanup**

Some rpm packages that are not needed for the system are:

```
Ash-0.3.8-8
Hotplug-2002_04_01-17
Python-2.2.2-26
Pyxf86config-0.3.5-1
Redhat-config-mouse-1.0.5-1
Redhat-logos-1.1.12-1
Rhpl-0.93-1
Usbutils-0.9-10
```

To remove this packages type:

```
rpm -e ash hotplug python pyxf86config redhat-config-mouse rhpl
usbutils redhat-logos
```

Now reduce the amount of programs that start up by going to /etc/init.d/ using the commands:

```
cd /etc/init.d
```

```
ls
```

Currently it should look like:

```
apmd        netfs
functions   random
halt        rawdevices
iptables    saslauthd
kdcrotate   single
killall     sendmail
keytable    sshd
kudzu       syslog
network
```

Run chkconfig to see what services are set to start at each run level using the following command:

```
chkconfig –list
```

The results should be similar to this:

```
kudzu       0:off   1:off   2:off   3:on    4:on    5:on    6:off
syslog      0:off   1:off   2:on    3:on    4:on    5:on    6:off
netfs       0:off   1:off   2:off   3:on    4:on    5:on    6:off
network     0:off   1:off   2:on    3:on    4:on    5:on    6:off
random      0:off   1:off   2:on    3:on    4:on    5:on    6:off
rawdevices  0:off   1:off   2:off   3:on    4:on    5:on    6:off
saslauthd   0:off   1:off   2:off   3:off   4:off   5:off   6:off
keytable    0:off   1:on    2:on    3:on    4:on    5:on    6:off
apmd        0:off   1:off   2:on    3:on    4:on    5:on    6:off
iptables    0:off   1:off   2:on    3:on    4:on    5:on    6:off
sshd        0:off   1:off   2:on    3:on    4:on    5:on    6:off
sendmail    0:off   1:off   2:on    3:on    4:on    5:on    6:off
```

The services that we can delete are kudzu, netfs, and saslauthd.. To remove these startup files, first stop any of the services that are running by the commands:

```
./kudzu stop
./netfs stop
```

Once the services are stopped, run chkconfig with the delete command to remove the scripts from the respective rc.d directories and then delete the files from the /etc/init.d directory as shown below:

```
chkconfig --del kudzu
chkconfig --del netfs
chkconfig --del saslauthd

rm kudzu netfs saslauthd
```

The directory should now look like:

```
apmd        ntp
functions   random
halt        rawdevices
iptables    sendmail
kdcrotate   single
killall     sshd
keytable    syslog
network
```

All the extraneous services need to be stopped for a couple of reasons. First, to reduce the needless resource consumption. The smaller number of services that are running the more resources are going to be available to accomplish the task of this machine, monitoring and capturing network traffic. Second, to reduce the risk to running servers that open and unused services may cause. Running fewer services reduces the footprint for threats. If there are only a handful of services running that are available from the network, there will be a reduced amount of ways to remotely compromise the machine

**Remove all unnecessary users and groups**

With an install of Red Hat linux, there are some users and groups that are created that will not be needed. The less users and groups the better as far as security is concerned.

Look at the /etc/passwd file to determine all of the users installed by default using the command:

```
cat /etc/passwd
```

Delete the default users that are not needed by executing, in order, the following commands:

```
userdel adm
userdel lp
userdel shutdown
userdel halt
userdel news
userdel uucp
userdel operator
userdel games
userdel gopher
userdel ftp
```

Now notice that the user rpm, which is used for the rpm management, is configured to have /bin/bash as it's login environment. The rpm user never needs to login so change this to /sbin/nologin by changing the line:

```
rpm:x:37:37::/var/lib/rpm:/bin/bash
```

To

```
rpm:x:37:37::/var/lib/rpm:/sbin/nologin
```

There are groups that can be deleted as well. To view the groups type:

```
cat /etc/group
```

Delete the default groups by executing the following commands, in order:

```
groupdel adm
groupdel lp
groupdel news
groupdel uucp
groupdel games
groupdel dip
groupdel users
```

Next, add a user account giving the person responsible for this system remote login capability, without giving root the same capability.

```
useradd sysadmin
passwd sysadmin
```

A second user needs to be created for the purpose of using chroot for the Snort process. Use the command:

```
useradd snort
```

Now we will set the immutable bit to prevent the files from being written to, deleted, and modifiedon the files that store user account information. This will effectively prevent access to all users, even root. This is accomplished by using the commands:

```
chattr +i /etc/password
chattr +i /etc/shadow
chattr +i /etc/group
chattr +i /etc/gshadow
```

If the files ever need to be modified, adding a new user for instance, execute the following command to remove the immutable bit. Remember that once the files have been modified, you need to reset the immutable bit to resume protection.

```
chattr -i /etc/password
chattr -i /etc/shadow
chattr -i /etc/group
chattr -i /etc/gshadow
```

**SUID and GUID programs**

Set UID and GUID programs can be dangerous because they will give the user that executed them the privilege of the owner or group of the file when they are executed. Typically the owner of the file is root, so when a user executes that binaryhe assumes root privileges within that process.
To find all of the SUID and GUID programs execute (single command expanded across two lines for formatting purposes of this paper):

```
find / -type f
\( -perm -04000 -o -perm -02000 \) -exec ls -l {} \;
```

The files that can have the SUID or GUID bit removed are:
`chage, gpasswd, wall, chfn, chsh, newgrp, write, usernetctl, ping, ping6, traceroute6, mount, umount, netreport`

Once files have been identified that can have the SUID or GUID bit removed, do so by executing the following commands:

```
chmod a-s /usr/bin/chage
chmod a-s /usr/bin/gpasswd
chmod a-s /usr/bin/wall
chmod a-s /usr/bin/chfn
chmod a-s /usr/bin/chsh
chmod a-s /usr/bin/newgrp
chmod a-s /usr/bin/write
chmod a-s /usr/sbin/usernetctl
chmod a-s /bin/ping
chmod a-s /usr/sbin/ping6
chmod a-s /usr/sbin/traceroute6
chmod a-s /bin/mount
chmod a-s /bin/umount
chmod a-s /sbin/netreport
```

Then, delete the following lines in /etc/inittab file:

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

This prevents multiple consoles being available by using the <alt>(<f2><f3><f4><f5><f6>) keys. In addition to increasing security, the deletion of these consoles frees up resources.

Remove the following files from the /root directory using the command “rm –f “ and then the file name.

anaconda-ks.cfg .

cshrc
install.log
install.log.syslog
.tcshrc
.Xresources

**Openssh Configuration**

Openssh, as stated earlier, will provide secure administration of the sensor. Any type of remote login or file copying will be done using ssh. Since Root will not be allowed to log in remotely, the administrator will have to login with his username and then su to root to make any changes to the system. Ssh will only use private/public key for user authentication. Public key authorization will prevent an attacker from being able to log in with a password as they will have to have the private key to be able to log in to the system. There will be no prompt for a password, If someone is able to connect to the system, but they do not have a private key that is a pair with the public key that is on the sensor, their ssh session will immediately be disconnected and an event will be logged in the log files.

In order to correctly setup the SSH daemon, the configuration file must be correctly modified. This file is located at /etc/ssh/sshd_config.

There will be no options that are commented out because if the options are in the file it is easier to turn them on than if the option is not in there at all. So the configuration file should be modified to look like this:

**/etc/ssh/sshd_config**

```
Port 22
Protocol 2
HostKey /etc/ssh/ssh_host_key
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key
KeyRegenerationInterval 3600
ServerKeyBits 768
SyslogFacility AUTHPRIV
LogLevel INFO
LoginGraceTime 15
PermitRootLogin no
StrictModes yes
RSAAuthentication no
PubKeyAuthentication yes
AuthorizedKeysFile      .ssh/authorized_keys
RhostsAuthentication no
IgnoreRhosts yes
RhostRSAAuthentication no
HostBasedAuthentication no
IgnoreUserKnownHosts no
PasswordAuthentication no
PermitEmptyPasswords no
ChallengeResponseAuthentication no
```

```
KerberosAuthentication no
KerberosOrLocalPasswd no
KerberosTicketCleanup no
KerberosTgtPassing no
PAMAuthenticationViaKbdInt no
X11Forwarding no
PrintMotd yes
PrintLastLog yes
KeepAlive yes
UseLogin no
UsePrivilegeSeparation yes
PermitUserEnvironment no
Compression yes
MaxStartups 10
Banner /etc/sshbanner
VerifyReverseMapping no
Subsystem   sftp  /usr/libexec/openssh/sftp-server
```

The following section describes the function of each setting in the SSH configuration file:

Port 22 – Binds to port 22 and the interface that gets the DHCP address
Protocol 2 – Sets the server to only use the SSH protocol version 2, which is more secure than version 1
HostKey /etc/ssh/ssh_host_key
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key
KeyRegenerationInterval 3600 – (in seconds) sets the time that the key will re-generate during a session.
ServerKeyBits 768 – specifies the number of bits to use when generating the RSA server key
SyslogFacility AUTHPRIV – What syslog facility to log to.
LogLevel INFO – What syslog log level
LoginGraceTime 15 – (in seconds) sets the time for if a users has not logged in by that the session disconnects
PermitRootLogin no – Does not allow the root user to remotely login
StrictModes yes – Specifies that the ssh server will check the file permissions and mode of the authorized key file in the user's home directory. The mode should be set to 400, which is read only for the owner, and the user should own the authorized_key file. The mode of the /home/sysadmin/.ssh should be 700, with sysadmin owning the folder.
RSAAuthentication no – does not allow RSA authentication, common in ssh version 1
PubKeyAuthentication yes – allows authentication using public and private keys.
AuthorizedKeysFile .ssh/authorized_keys – sets the authorized key file.
RhostsAuthentication no – Does not allow Rhost authentication, which is trust based authentication between system, inherently insecure.
IgnoreRhosts yes – again related to Rhost authentication, will ignore any rhost file in a users directory

RhostRSAAuthentication no – prevents authentication using a combination of RSA and Rhost authentication
HostBasedAuthentication no – Similar to RhostRSAAuthentication but applies to ssh version 2, uses a combination of a public key with rhost authentication.
IgnoreUserKnownHosts no – relates to host based security.
PasswordAuthentication no – does not allow users to use passwords to login with, this system is only using dsa public and private keys for authentication.
PermitEmptyPasswords no – again does not allow any passwords to be blank, does not apply because this system does not use passwords
ChallengeResponseAuthentication no – prevents any type of challenge and response type authentication.
KerberosAuthentication no – prevent Kerberos authentication.
KerberosOrLocalPasswd no – if Kerberos is not available, it would use /etc/passwd to authenticate the user.
KerberosTicketCleanup yes – default, but does not apply, would destroy a user's Kerberos ticket on logout
KerberosTgtPassing no – does not allow a Kerberos ticket to be passed to the sensor
PAMAuthenticationViaKbdInt no -
X11Forwarding no – Xwindows is not running on this system, redundant but layered security, will not allow Xwindows to be forwarded over an ssh session.
PrintMotd yes – prints the message of the day when the sysadmin user logs in.
PrintLastLog yes – prints the date, time, and from what IP that the sysadmin last logged in from. This should always be the same because the sysadmin should always come from the same management station
KeepAlive yes – a way to monitor a connection so that a session can be shutdown if the network goes down or if a users abruptly disconnects.
UseLogin no – set so that ssh does not use the "login" utility to login the user.
UsePrivilegeSeparation yes – a new feature added to run the ssh session with the privileges that the user logs in with. The daemon is initial run as root, so that it can bind to port 22 and when a user logins, the ssh session is dropped to the privilege of that user.
PermitUserEnvironment no – prevents a users from specify environmental settings in /home/username/.ssh/environment.
Compression yes – turns compression on for network transfer
MaxStartups 10 – allows only 10 sessions to be running that do not have authenticated users. Ssh will only allow more users to authenticate, once either the grace period of 15 seconds have gone by or the user successfully logs in. Prevent ssh from starting some many processes that the system can not do anything else.
Banner /etc/sshbanner – presents a warning banner before the user logs in. For legal reasons.
VerifyReverseMapping no – tries to valid the host name against the IP address, left at default of no.
Subsystem sftp /usr/libexec/openssh/sftp-server – sets up the subsystem sftp, secure ftp.

Be sure to set the banner file "/etc/banner" to say something approved by the legal department. It should appropriately identify the system and give access restrictions. For example, it may say that the system belongs to Information Security and that all other personnel are not allowed access.

Once the sshd_config file has been configured, save the file and restart the ssh daemon. This is done by restarting the service using the startup script in /etc/init.d using the following command:

```
/etc/init.d/sshd restart
```

Now that the daemon is setup for ssh, the public key generated for the sysadmin user needs to be moved to the sensor. Create a directory called .ssh in /home/sysadmin/, set the mode to read only for the owner, and move the public key into that directory with a filename of authorized_keys and a mode of read only for the owner as shown below:

```
mkdir /home/sysadmin/.ssh
chmod 700 /home/sysadmin/.ssh
mv /mnt/floppy/id_dsa.pub /home/sysadmin/.ssh/authorized_keys
chmod 400 /home/sysadmin/.ssh/authorized_keys
```

**NTP Setup and Configuration.**

Network Time Protocol is used to keep different computers all set to the same time. The reason for NTP on the sensor is to make sure that it easier to correlate events that show up on systems. This paper is going to assume that there is already a best practices NTP setup in place at the facility that this sensor is going to be deployed.

The configuration files for NTP that need to be configured are "/etc/ntp/step-tickers" and "/etc/ntp.conf". Each location will have time servers deployed and defined. Assume for the purposes of this paper that the deployment location for this sensor has time servers located at 192.168.0.123, 192.168.1.123, and 192.168.2.123.

The file /etc/ntp/step-tickers must contain the ip addresses of the time servers. Change the files defined as shown below:

**`/etc/ntp/step-tickers`** :

```
192.168.0.123
192.168.1.123
192.168.2.123
```

**`/etc/ntp.conf`**

```
driftfile /etc/ntp/drift
```

```
server 127.127.1.1 #local clock
fudge 127.127.1.1 stratum 10

server 192.168.0.123
server 192.168.1.123
server 192.168.2.123

restrict 192.168.0.123 no modify, noquery
restrict 192.168.0.123 no modify, noquery
restrict 192.168.0.123 no modify, noquery

restrict default nomodify
restrict 127.0.0.1
```

The reasons for the file modifications are shown below:Driftfile /etc/ntp.drift – information is stored in this file on how much the local clock drifts from the timeservers. Therefore, if the timeservers become became unavailable, the system uses this information to stay fairly close in sync.

```
restrict 192.168.0.123 nomodify, noquery
restrict 192.168.1.123 nomodify, noquery
restrict 192.168.2.123 nomodify, noquery
```

These are the servers that the system syncs against, nomodify and noquery prevents the servers from accessing or controlling the sensor.

```
server 192.168.0.123
server 192.168.1.123
server 192.168.2.123
```

These are the internal server ip addresses of the NTP Servers.

Restrict default nomodify – the default policy is that there are no servers that can remotely modify the configuration.
Restrict 127.0.0.1 – there is no restriction for the local server's operation.
fudge 127.127.1.0 stratum 10
server 127.127.1.0 #local clock – this code sets the local clock as stratum 10, so if the ntp servers are not available, it will keep in sync off of the local time.

Now stop and restart the NTP service using the following commands:

```
/etc/init.d/ntpd restart
```

### Sendmail configuration

When the system initially boots up in the production environment, it is basically an unknown entity to the management console. In order for it to be setup, the administrator needs to know its IP address. Sendmail will be utilized once the

system is booted up to send an email to the administrator's mailbox with its IP address. To prevent anyone from accessing Sendmail from a remote IP address, it will only be bound to the loopback address.

To configure sendmail, copy "/etc/mail/sendmail.mc" to "/etc/mail/sendmail.mc.original" as a backup in case anything goes wrong. Then, edit sendmail.mc. Next, run the m4 command to save the new configuration. Finally remove the m4 rpm package. This sequence is shown below:

```
cp /etc/mail/sendmail.mc /etc/mail/sendmail.mc.original
```

Edit sendmail.mc with the syntax in the appendix and then remove the m4 and sendmail-cf package:

```
m4 /etc/mail/sendmail.mc > /etc/sendmail.cf
rpm –e m4 sendmail-cf
```

Now there needs to be a notification script setup so that when the sensor is booted up, it will fire off an email to the administrator so that he/she can manage it. Create a directory in "/etc" called "notify". The script will be named "notify.sh" and will be called by rc.local. The reason why it is put in rc.local is because rc.local is called after all other startup scripts have been executed.

```
mkdir /etc/notify
```

**/etc/notify/notify.sh**

```
ifconfig | grep -A 4 eth0 | awk '/inet/ { print $2 } ' | sed -e s/addr:// |\
mail -s SnortSensorAlive sysadmin@ten0808080.com
```

Make sure notify.sh is set to a mode of 500:

```
chmod 500 /etc/notify/notify.sh
```

Edit /etc/rc.local and add the following line:

```
/etc/notify/notify.sh
```

This will send an email to sysadmin@ten0808080.com with a subject of "SnortSensorAlive". The only contents of the email will be the ip address that the sensor obtained from the dhcp server when the system is started up.

### Tcp Wrappers configuration

Tcp wrappers provide a way to restrict, by ip addresses, access to certain daemons and also provides a logging facility for unauthorized attempts. In this configuration, the only daemons that this will apply to will be the ssh and sendmail daemon. The two files that pertain to tcp wrappers are "/etc/hosts.allow

" and "/etc/hosts.deny". Hosts.allow contains the IP addresses that are allowed to access specific daemons, everything else is denied. Hosts.deny contains the IP addresses that are denied access to a daemon, everything else is allowed. Hosts.allow is read before hosts.deny, so if a host is in hosts.allow it will be allowed, if it is not, and is in hosts.deny, it is denied access. If the host is not listed in either file, it is allowed access.

The proper configuration for hosts.allow and hosts.deny is

hosts.deny
```
ALL: ALL
```

This states all host are denied access to any daemon, the idea of fail closed.

hosts.allow
```
SSHD: 192.168.10.100
SENDMAIL: 127.0.0.1
```

This states the host at 192.168.10.100 (which is a management station) can access to the ssh daemon and the loopback address can communicate with the sendmail daemon.

**Snort Installation and configuration**

On the staging server, download the source code to snort at http://www.snort.org/dl/snort-2.0.0.tar.gz. Snort v2.0 is the latest version at the time of this writing. Download the pgp signature at http://www.snort.org/dl/snort-2.0.0.tar.gz.asc. Lastly, download the public key to be able to verify the contents at http://pgp.mit.edu:11371/pks/lookup?op=get&search=0x1946E4A1.
At the staging server execute:

```
wget http://www.snort.org/dl/snort-2.0.0.tar.gz
wget http://www.snort.org/dl/snort-2.0.0.tar.gz.asc
wget http://pgp.mit.edu:11371/pks/lookup?op=get\&search=0x1946E4A1 -O
snort.pgp
```

Import the Snort's public key into gpg and verify the source code by the commands:

```
gpg --import snort.pgp
gpg --verify snort-2.0.0.tar.gz.asc snort-2.0.0.tar.gz
```

It should read:

```
gpg: Good signature from "Snort.org releases
<releases@snort.org>"
```

The libpcap package will need to be installed on both the staging server and the snort sensor

Once the code has been verified, extract it and compile the source code. It is important that the same version of libpcap is installed on the staging server and the snort sensor before proceeding. Use the following commands to install Snort:

```
tar –zxvf snort-2.0.0.tar.gz
cd snort-2.0.0
./configure
make
```

There should now be a file called snort in the snort-2.0.0/src directory.
On the snort sensor, there needs to be a directory structure setup so that Snort can be chroot'ed in to that new directory structure. First create "/var/snort", then change in to that directory and create 5 more directories named dev, etc, rules, sbin, var. Change the owner of the var directory to snort.root. Create the logging directories for snort under the var directory and make sure the owner is setup the same: snort.root. Now change into "/var/snort/dev" and create the special devices null and zero. The following are the commands for the above description:

```
mkdir /var/snort
cd /var/snort
mkdir dev etc rules sbin var
mkdir var/log var/log/snort
chown –R snort.root var
cd /var/snort/dev
mknod null c 1 3
mknod zero c 1 5
chmod –R 750 /var/snort
```

Now copy the snort binary from the staging server to the snort sensor using this command:

```
cp /mnt/floppy/snort /var/snort/sbin/
```

There will not be a start up file for snort because it will be run on an as needed basis. This paper will not cover all of the commands for snort. Howver, there is a great description at http://www.snort.org/docs/writing_rules/ . Generally, Snort needs the following commands to start:

```
var/snort/sbin/snort -t /var/snort -l /var/snort/var/log/snort  \
-u snort
```

-t identifies the root of the chrooted file system.
-l identifies the logging directory for the snort process, that is why snort had to be the owner, so it would have permission to that directory.
-u identifies what user that the snort process will run in.

## Iptables configuration

The iptables script will be setup as a host level firewall. This file will replace the default /etc/init.d/iptables file that Red Hat installs.

The iptables configuration file will be in "/etc/init.d" and should read as follows:

**/etc/init.d/iptables**

```
#!/bin/sh
#
# Startup script to implement /etc/sysconfig/iptables pre-defined rules.
#
# chkconfig: 2345 08 92
#
# description: Automates a packet filtering firewall with iptables.
#

# Source 'em up
. /etc/init.d/functions


if [ ! -x /sbin/iptables ]; then
        exit 0
fi


start() {
         echo -n $"Applying iptables firewall rules: "
         SECURE_IFACE="eth0"

         LO_IFACE="lo"
         LO_IP="127.0.0.1"

         NTP1="192.168.0.123"
         NTP2="192.168.1.123"
         NTP3="192.168.2.123"

         SMTP="192.168.25.25"
         DNS="192.168.25.53"
         NTPSERVERS=( $NTP1 $NTP2 $NTP3 )
         IPTABLES="/sbin/iptables"
         #
         # Set default policies for the INPUT, FORWARD and OUTPUT chains to Drop
any
         # packets that
         # are not explicitly defined.
         #
         $IPTABLES -F
         $IPTABLES -X
         $IPTABLES -P INPUT DROP
         $IPTABLES -P OUTPUT DROP
         $IPTABLES -P FORWARD DROP


         #
         # Creates a new IPTABLE chain for the portknocking daemon to add rules
and
         # clear so that management sessions can be created.
         #
         $IPTABLES -N ssh_rule

         #
         # This sets the input chain to allow any type of communication on
         # the loopback interface, allows traffic back in that was initiated
         # from this machine, it will then log the packet information and
         # drop the packet
         #
```

```
        $IPTABLES -A INPUT -p ALL -s $LO_IP -d $LO_IP -j ACCEPT
        $IPTABLES -A INPUT -p TCP -i $SECURE_IFACE --dport 22 --syn -j ssh_rule
        $IPTABLES -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
        $IPTABLES -A INPUT -p ALL -j LOG --log-prefix "DENY " --log-level info
        $IPTABLES -A INPUT -j DROP


        #
        # OUTPUT chain
        # This allows for the loopback interface to have no restrictions, the
sensor to
        # contact the Time Servers for time synchronization, and finally to reply
        # if a session is already setup.
        #
         $IPTABLES -A OUTPUT -p ALL -s $LO_IP -d $LO_IP -j ACCEPT
        for NTPS in "${NTPSERVERS[@]}"
        do
        $IPTABLES -A OUTPUT -p UDP -o $SECURE_IFACE -d $NTPS --dport 123 -j
ACCEPT
        done
        $IPTABLES -A OUTPUT -p TCP -o $SECURE_IFACE -d $SMTP --dport 25 -j ACCEPT
        $IPTABLES -A OUTPUT -p UDP -o $SECURE_IFACE -d $DNS --dport 53 -j ACCEPT
        $IPTABLES -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
        $IPTABLES -A OUTPUT -j DROP

        touch /var/lock/subsys/iptables
}

stop() {
        /sbin/iptables -F
        /sbin/iptables -X
        /sbin/iptables -P INPUT ACCEPT
        /sbin/iptables -P OUTPUT ACCEPT

        echo -n $"Resetting built-in chains to the default ACCEPT policy:"
        echo
        rm -f /var/lock/subsys/iptables
}

case "$1" in
  start)
        start
        ;;

  stop)
        stop
        ;;

  restart)
        # "restart" is really just "start" as this isn't a daemon,
        #  and "start" clears any pre-defined rules anyway.
        #  This is really only here to make those who expect it happy
        start
        ;;
condrestart)
       [ -e /var/lock/subsys/iptables ] && start
        ;;

  status)
        tables=`cat /proc/net/ip_tables_names 2>/dev/null`
        for table in $tables; do
                echo $"Table: $table"
                iptables -t $table --list
        done
        ;;


  *)
        echo $"Usage: $0 {start|stop|restart|condrestart }"
        exit 1
esac
```

```
exit 0
```

Once this file is in place and the sensor is rebooted or started, the firewall will be setup.

**Port Knocking Configuration**

Download the code from onto the staging server.
http://mkweb.bcgsc.bc.ca/portknocking/distribution/portknocking-0.1.tgz

and download the md5 has from
http://mkweb.bcgsc.bc.ca/portknocking/distribution/portknocking-0.1.md5

Verify that the md5 hash is the same as what is listed in the portknocking-0.1.md5 file:

```
md5sum portknocking-0.1.tgz
c1a47294630325258cde93e90cfbbe62  portknocking-0.1.tgz
cat portknocking-0.1.md5
c1a47294630325258cde93e90cfbbe62  portknocking-0.1.tgz
```

Once the values are verified as equal, it should be safe to run the code.Although it is possible for the hacker to upload a md5sum file for a compromised file, the programs are in perl making thema lot easier to read and pick out any questionable code.

Copy the knockdaemon file to the sensor and the knockclient to the management station. The first thing that needs to be modified in both knockdaemon and knockclient is to change the location of the perl interpreter. The first line reads:

```
#!/usr/local/bin/perl
```

This should be changed to read:

```
#!/usr/bin/perl
```

The File::Tail, Math::VecStat, and Crypt::CBC perl modules will also need to be installed on the sensor. On the management station, for the client, Math::VecStat and Crypt::CBC will need to be installed. To download the modules execute these commands on the staging server:

```
wget http://search.cpan.org/src/ASPINELLI/Math-VecStat-
    0.08/VecStat.pm
wget http://search.cpan.org/src/LDS/Crypt-CBC-2.08/CBC.pm
wget http://search.cpan.org/src/MGRABNAR/File-Tail-0.98/Tail.pm
```

Transfer VecStat.pm, CBC.pm, and Tail.pm, to the sensor
Place the files in the following locations:

VecStat.pm -> /usr/lib/perl5/5.8.0/Math/

```
mv VecStat.pm /usr/lib/perl5/5.8.0/Math/
```

CBC.pm -> /usr/lib/perl5/5.8.0/Crypt/

```
mkdir /usr/lib/perl5/site_perl/5.8.0/Crypt
mv CBC.pm /usr/lib/perl5/site_perl/5.8.0/Crypt
```

Tail.pm -> /usr/lib/perl5/5.8.0/File

```
mv Tail.pm /usr/lib/perl5/5.8.0/File/
```

On the management station, transfer CBC.pm and VecStat.pm to the proper locations:

```
mv VecStat.pm /usr/lib/perl5/5.8.0/Math/
mkdir /usr/lib/perl5/site_perl/5.8.0/Crypt
mv CBC.pm /usr/lib/perl5/site_perl/5.8.0/Crypt
```

The Crypt::Blowfish module will have to be downloaded and compiled on the staging server using the commands

```
Wget http://search.cpan.org/CPAN/authors/id/D/DP/DPARIS/Crypt-
     Blowfish-2.09.tar.gz

tar -zxvf Crypt-Blowfish-2.09.tar.gz
cd Crypt-Blowfish-2.09
perl Makefile.pl
make
make test
make install
```

Transfer the following files from the staging server to the sensor:

```
/usr/lib/perl5/site_perl/5.8.0/i386-linux-thread-
multi/auto/Crypt/Blowfish/Blowfish.so

/usr/lib/perl5/site_perl/5.8.0/i386-linux-thread-multi/Crypt/Blowfish.pm
```

On the sensor create the following directories:

```
/usr/lib/perl5/site_perl/5.8.0/i386-linux-thread-multi/auto
/usr/lib/perl5/site_perl/5.8.0/i386-linux-thread-multi/auto/Crypt
/usr/lib/perl5/site_perl/5.8.0/i386-linux-thread-multi/auto/Crypt/Blowfish
/usr/lib/perl5/site_perl/5.8.0/i386-linux-thread-multi/Crypt/
```

Copy the files from the staging server into the proper directories on the sensor:

```
Blowfish.so -> /usr/lib/perl5/site_perl/5.8.0/i386-linux-thread-
multi/auto/Crypt/Blowfish
Blowfish.pm -> /usr/lib/perl5/site_perl/5.8.0/i386-linux-thread-multi/Crypt/
```

This same process needs to be performed on the Management machine as well.

Modifications need to be made to the knockdaemon file. Originally it was setup to allow for a time to be set to close the port when finished. The modifications will consist of either sending a time value of 0, which will add a rule that opens access to the port, or any value other than 0 up to 255, which will close the port. The actual file will be listed in the appendix. Also, it will be configured to use IPTABLES instead of ipchains.

To configure the client and daemon, there is a line in each:

```
use constant KEY  => "knock"; # keyphrase for the encryption
```

"knock" is currently the password for the blowfish encryption, change the word "knock" to something in both files, so that it uses a different password than what is the default. For instance:

```
use constant KEY  => "dR0w5s4P"; # keyphrase for the encryption
```

Make sure to do this in BOTH files, knockclient and knockdaemon, otherwise the daemon will not be able to decrypt the message from the client.

Once the file knockdaemon has been configured properly, make a directory called "/usr/local/knockdaemon" and move the file into the directory. Make sure that the permissions are set to 500 and owned by root.root.

```
mkdir /usr/local/knockdaemon
mv knockdaemon /usr/local/knockdaemon
```

Now add to rc.local the following line:

```
/usr/local/knockdaemon/knockdaemon -f /var/log/messages &
```

To use the knockclient to access sensor execute:

```
./knockclient -c 192.168.10.100 -r 192.168.22.46 -p 22 -t 0

-c is the clients address
-r is the ip address of the sensor, obtained from the email sent
out after bootup
-p is the port to open
-t whether to open or close port
```

Port knocking allows the sensor to essentially be invisible to the network. SSH, the only available service will only be available once the knockclient has sent the correct series of port attempts. The knockdaemon will the open the port after the series of ports are encrypted with blowfish encryption, requiring a secret key to be known by the client. There is also a CRC redundancy check to help protect the system., Finally, even if the attack was able to send the correct port atttemps to open a port from themselves, there are rules in both tcpwrappers and iptables

that would prevent them from getting access to be able to authenticate to the ssh daemon.

**Tripwire Configuration**

Tripwire will be configured to monitor specific operating system and application files for any type of modifications, deletions, or additions. If it detects any modifications, it will alert the administrator. The administrator should be the only one making any changes to the system, so if it detects something out of the ordinary, it should raise an immediate red flag.

Run the after rpm install file," /etc/tripwire/twinstall.sh"

```
/etc/tripwire/twinstall.sh
```

In this script, it is going to ask for the site keyfile passphrase and the local keyfile passphrase. The site key is used to sign the configuration file and the policy file. The local key is used to encrypt the local database.

Copy the /etc/tripwire/twpol.txt file listed in the appendix to the sensor and update the create the policy file (wrapped for formatting purposes):

```
cp /mnt/floppy/twpol.txt /etc/tripwire
/usr/sbin/twadmin --create-polfile -S site.key
/etc/tripwire/twpol.txt
```

It will ask for the site key's passphrase because it has to encrypt the new policy file.

Now initialize tripwire, run the following command:

```
/usr/sbin/tripwire --init
```

It will ask for the local key's passphrase to encrypt the database file.
The default tripwire configuration file will be used.

Once the tripwire is configured, there is a default script in "/etc/cron.daily/" that will run tripwire on a daily basis. If there is a need for faster reporting, setup a crontab or move the script into the "/etc/cron.hourly".

Copy the policy, configuration, and database file to floppy or cdrom media. If the system is ever compromised, it is possible that those files have also been compromised as well. Anytime that the database is updated be sure to move a copy of the system, again either to cdrom or floppy media. Also, remove the *.txt files out of the /etc/tripwire directory. These files contain unencrypted configuration for tripwire.

## Ongoing maintenance

### Patch maintenance

Patch maintenance is very important because the majority of the exploits out currently are based on vulnerabilities that already have patches released for them.

Since this system will not be online all the time, it will be updated by transferring the updates from the management system. This is done by ftp'ing to ftp://rpmfind.net/linux/redhat/updates/9/en/os from the management system , downloading all of the latest updates, and burn them to a CD. Once created, the CD can be transferred to the Snort Sensor, mounted, and executed using , the "Freshen" command in the RPM application. The Freshen command will on update the existing applications that are out of date. The command is shown below:

> rpm –Fvh *.rpm .
>
> Note: I use rpmfind.net because ftp.redhat.com seems to be busy more often than not. Rpmfind.net is an official mirror of ftp.redhat.com: http://www.redhat.com/download/mirror.html.

This is the same method used for updating the packages of the initial install. If there is ever a critical patch that is released and the system is online, the rpm can be copied via scp to the sensor and updated immediately.

Updates for snort are available from http://www.snort.org
Any updates to snort will involve download the source code and compiling it on a separate machine. The binary will be moved over to the snort sensor as needed.

### Mailing Lists and Web Sites

Mailing lists and web sites provide up to date information about new and existing vulnerabilities in the operating system and software deployed. Some common sites and list to visit are listed below.

> http://www.redhat.com/mailman/listinfo/redhat-watch-list - "Redhat's list regarding critical bug fixes and security issues in Redhat Linux"
> http://www.securityfocus.com/subscribe - SecurityFocus hosts multiple mailing lists for OS'. Bugtraq and Linux Focus are two good mailing lists here.

Websites:

http://packetstorm.linuxsecurity.com/ - Good general information for linux security, including tools and exploits.
https://rhn.redhat.com/errata/rh9-errata.html - Redhat Linux 9 advisories for security, bug fixes, and enhancements
http://www.securityfocus.com/ - Another good general security site that contains registration for the bugtraq mailing list.
http://www.snort.org - Snort's main web site. Contain links to the source code, rule sets, and any information relating to vulnerabilities.

### Snort Rule Maintenance

The snort latest rule set will be downloaded from http://www.snort.org/dl/rules/snortrules-stable.tar.gz and compared against the installed rules. The whole rule set will not be utilized in every deployment as each case will be unique. There maybe a case where monitoring of HTTP will be needed or looking for specific SMTP traffic. Any new published rule may be added as need, but in most cases custom rules will be written.

### Review the Logs

The log files generated by the system are there for a reason. If the system has been rebooted or unplugged and restarted, an event will be logged in the log files. By using Logcheck, a log consolidator, the amount of information can be reduced to specific log entries setup by using specific keywords. A daily report may be mailed to root's mailbox for daily review. This does not take the place of reviewing the actual log files for any information that may not be covered in the keywords that are defined, but alerts the administrator that the logs need to be reviewed. Tripwire will also generate an email report that will notify the administrator of any changes, additions, or deletion to the files and directories that are being monitored.

### Scheduled Vulnerability Assessment

Once the system is in place an ongoing assessment will take place to check for any compromise or malfunction. A subset of the assessment preformed in the "Check your configuration" will be preformed when the unit is in the field. Port scanning using nmap to verify that there are no new listening ports.

## Check your configuration

### Verify that port knocking is working

Turn on and off rule with time settings:

(Management system)

```
./knockclient –c 192.168.10.100 –r 192.168.20.35 –p 22 –t 0
```

(Sensor system)
```
iptables –L
```
Verify that there is a rule that allows 192.168.10.100 to access port 22

(Management system)
```
./knockclient –c 192.168.10.100 –r 192.168.20.35 –p 22 –t 1
```
(Sensor system)
```
iptables –L

Chain ssh_rule (1 references)
target     prot opt source               destination
ACCEPT     tcp  --  192.168.10.100        anywhere           tcp
dpt:ssh
```

Verify that there are no rules listed on the ssh_rule chain.

Management to Sensor with wrong password check. In the knockclient, change the line:

```
use constant KEY  => "password"; # keyphrase for the encryption
```

Make sure line in knockdaemon is something different. On the management station execute:

```
./knockclient –c 192.168.10.100 –r 192.168. –p 22 –t 0
```

At the sensor, see if it added a rule to iptables:

```
iptables –l

Chain ssh_rule (1 references)
target     prot opt source               destination
```

There should not be a rule under the ssh_rule chain

**Check services that are running**

Verify what services are running by executing and what daemons are listening for network traffic using the following command:

```
ps –aux
```

The following is an example of the response:

```
UID        PID  PPID  C STIME TTY          TIME CMD
root         1     0  0 Jun15 ?        00:00:04 init
root         2     1  0 Jun15 ?        00:00:00 [keventd]
root         3     1  0 Jun15 ?        00:00:00 [kapmd]
root         4     1  0 Jun15 ?        00:00:00 [ksoftirqd_CPU0]
root         9     1  0 Jun15 ?        00:00:00 [bdflush]
root         5     1  0 Jun15 ?        00:00:00 [kswapd]
```

```
root         6     1  0 Jun15 ?        00:00:00 [kscand/DMA]
root         7     1  0 Jun15 ?        00:00:00 [kscand/Normal]
root         8     1  0 Jun15 ?        00:00:00 [kscand/HighMem]
root        10     1  0 Jun15 ?        00:00:00 [kupdated]
root        11     1  0 Jun15 ?        00:00:00 [mdrecoveryd]
root        15     1  0 Jun15 ?        00:00:00 [kjournald]
root        73     1  0 Jun15 ?        00:00:00 [khubd]
root       645     1  0 Jun15 ?        00:00:00 [kjournald]
root       646     1  0 Jun15 ?        00:00:00 [kjournald]
root       647     1  0 Jun15 ?        00:00:00 [kjournald]
root       895     1  0 Jun15 ?        00:00:00 /sbin/dhclient -1 -q -lf
/var/lib/dhcp/dhclient-eth0.leases -pf /
root       938     1  0 Jun15 ?        00:00:00 syslogd -m 0
root       942     1  0 Jun15 ?        00:00:00 klogd -x
root       978     1  0 Jun15 ?        00:00:00 /usr/sbin/apmd -p 10 -w 5 -W -P
/etc/sysconfig/apm-scripts/apmscr
root       989     1  0 Jun15 ?        00:00:00 /usr/sbin/sshd
ntp       1002     1  0 Jun15 ?        00:00:06 [ntpd]
root      1035     1  0 Jun15 ?        00:00:00 /usr/bin/perl
/usr/local/knockdaemon/knockdaemon -f /var/log/messages
root      1137     1  0 Jun15 ?        00:00:00 sendmail: accepting connections
smmsp     1146     1  0 Jun15 ?        00:00:00 sendmail: Queue runner@01:00:00
for /var/spool/clientmqueue
root      1158     1  0 Jun15 ?        00:00:00 crond
root      1290     1  0 Jun15 tty1     00:00:00 /sbin/mingetty tty1
root      1544   989  0 21:12 ?        00:00:00 /usr/sbin/sshd
sysadmin  1546  1544  0 21:12 ?        00:00:00 /usr/sbin/sshd
sysadmin  1547  1546  0 21:12 pts/0    00:00:00 -bash
root      1581  1547  0 21:12 pts/0    00:00:00 su -
root      1582  1581  0 21:12 pts/0    00:00:00 -bash
root      1640  1582  0 21:16 pts/0    00:00:00 ps -ef
```

This is what the ps –ef output should look like if someone has a ssh session open.

```
netstat –an

Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN
udp        0      0 0.0.0.0:68              0.0.0.0:*
```

With the output from netstat, anything that is **listening** besides tcp port 22 and tcp port 25 bound to 127.0.0.1, should be held as suspicious. Tcp port 22 is the default port that the ssh daemon listens on, and the sendmail daemon has to listen on port 25 to receive email for the local host.

### Verify that there are no open ports

Scan the system with nmap to see that there are no open ports:

```
nmap -sT -P0 -p1-10,65530-65535 192.168.20.356

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
All 65535 scanned ports on  (192.168.20.35) are: filtered

Nmap run completed -- 1 IP address (1 host up)

nmap -sU -P0 -p1-10,65530-65535 192.168.20.35
```

```
Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on  (192.168.20.35):
Port       State       Service
1/udp      open        tcpmux
2/udp      open        compressnet
3/udp      open        compressnet
4/udp      open        unknown
5/udp      open        rje
6/udp      open        unknown
7/udp      open        echo
8/udp      open        unknown
9/udp      open        discard
10/udp     open        unknown
....
....
....
65530/udp  open        unknown
65531/udp  open        unknown
65532/udp  open        unknown
65533/udp  open        unknown
65534/udp  open        unknown
65535/udp  open        unknown

Nmap run completed -- 1 IP address (1 host up)
```

The TCP scan should report all of the ports as filtered because the packets are being dropped and there is no response back from the sensor. The UDP scan should report all of the ports as open because since the packets are being dropped there is no response back from the host. The way udp works is since there is no response sent back to the sender, nmap will assume that it was received and report it as open. If the udp packets were not being dropped, the sensor would send back an ICMP port unreachable, Type 3, Code 3. If nmap receives the ICMP Type 3 Code 3 then it will report the port as closed.

If there are any ports that are reported as open on the TCP scan, re-check the configuration. By the iptables policies, there should be no opened ports. The only time an open port should exist is if the knockclient submits the correct packets to open a port and that port has a listening daemon on it.

**Check that snort is properly logging**

Boot the sensor with it connected to a hub, if a switch is used, unless the port is a spanned port, there will not be any traffic going to it unless it is multicast or broadcast traffic. Start the snort process for a few minutes and then kill it. Check the log file to make sure it caught all of the traffic.

```
/var/snort/sbin/snort -t /var/snort -bl /var/snort/var/log/snort
\
```

```
-u snort

<crtl c>

/var/snort/sbin/snort –devr /var/log/snort/<name of log file>
```

**Verify management traffic is encrypted**

Start snort logging in binary format. Run the knockclient on the management station to open access to the ssh daemon. Start a ssh session to the sensor. Once logged on to the system, exit out and run the knockclient to close the access.

On the Snort sensor:

```
/var/snort/sbin/snort -t /var/snort -bl /var/snort/var/log/snort
\
-u snort 'port 22'
```

On the management station:

```
./knockclient –c 192.168.100.23 –r 192.168.20.35 –p 22 –t 0
ssh –i .ssh/sysadmin 192.168.20.35
```

(there should be a remote session established now)

```
exit
```

(back on the management server)

```
./knockclient –c 192.168.100.23 –r 192.168.20.35 –p 22 –t 255
```

On the Snort Sensor:

<ctrl c> to stop snort logging

```
/var/snort/sbin/snort –devr /var/snort/var/log/snort/<name of logfile>   |less
```

Scroll through this file, if there are any characters that resemble any legible text, the connection is not encrypted.

**Check alerting function**

Test cases:

Reboot the system and verify that it shows up in the logs
Try to log in without a private key to the ssh, verify that it shows up in the logs:
Run knockclient to open the port:

```
./knockclient –c 192.168.10.100 –r 192.168.20.35 –p 22 –t 0
```

Now attempt to log into ssh without using the private key:

```
ssh sysadmin@192.168.20.35
```

On the sensor check in /var/log/messages to make sure an invalid log in attempted happened:

```
SnortSensor su(pam_unix)[1448]: authentication failure;
logname=sysadmin uid=500 euid=0 tty= ruser=sysadmin rhost=
user=root
```

Add a file in a monitored directory, run tripwire and verify that it is alerting:

```
touch /usr/local/knockdaemon/thisisatest
tripwire --check
```

## References


Matt Kettler - chroot snort
http://marc.theaimsgroup.com/?l=snort-users&m=105059656608683&w=2

Red Hat's Reference guide to configuring tripwire
http://www.redhat.com/docs/manuals/linux/RHL-9-Manual/ref-guide/ch-tripwire.html

Mourani, Gerhard. Securing and Optimizing Linux: Red Hat Edition, June 7, 2000
http://www.openna.com/products/books/securing-optimizing-linux/solrhe.htm

Krzywinski, Martin. Port Knocking — Network Authentication Across Closed Ports, Sysadmin Magazine June 2003 • Volume 12 • Number 6

Snort Documentation – http://www.snort.org

Nmap Documentation - http://www.insecure.org/nmap/nmap_documentation.html

Andreasson, Oskar. Iptables Tutorial 1.1.11
http://www.netfilter.org/documentation/tutorials/blueflux/iptables-tutorial.html

Krzywinski, Martin. Port Knocking home site
http://mkweb.bcgsc.bc.ca/portknocking/

Symantec's Security Site, Sendmail Trojan Horse Vulnerability
http://securityresponse.symantec.com/avcenter/security/Content/5921.html

Houston's Linux User Group, Latest libpcap & tcpdump sources from tcpdump.org contain a trojan.
http://www.hlug.org/trojan/


## Appendix

### Buffer Overflows

http://www.hackinthebox.org/article.php?sid=7952
http://destroy.net/machines/security/P49-14-Aleph-One

http://destroy.net/machines/security/buffer.txt

## Sendmail.mc

## /etc/mail/sendmail.mc

```
divert(-1)dnl
dnl #
dnl # This is the sendmail macro config file for m4. If you make changes to
dnl # /etc/mail/sendmail.mc, you will need to regenerate the
dnl # /etc/mail/sendmail.cf file by confirming that the sendmail-cf package is
dnl # installed and then performing a
dnl #
dnl #     make -C /etc/mail
dnl #
include(`/usr/share/sendmail-cf/m4/cf.m4')dnl
VERSIONID(`setup for Red Hat Linux')dnl
OSTYPE(`linux')dnl
dnl #
dnl # Uncomment and edit the following line if your outgoing mail needs to
dnl # be sent out through an external mail server:
dnl #
dnl define(`SMART_HOST',`mailserver.ten0808080.com’)
dnl #
define(`confDEF_USER_ID',``8:12'')dnl
define(`confTRUSTED_USER', `smmsp')dnl
dnl define(`confAUTO_REBUILD')dnl
define(`confTO_CONNECT', `1m')dnl
define(`confTRY_NULL_MX_LIST',true)dnl
define(`confDONT_PROBE_INTERFACES',true)dnl
define(`PROCMAIL_MAILER_PATH',`/usr/bin/procmail')dnl
define(`ALIAS_FILE', `/etc/aliases')dnl
dnl define(`STATUS_FILE', `/etc/mail/statistics')dnl
define(`UUCP_MAILER_MAX', `2000000')dnl
define(`confUSERDB_SPEC', `/etc/mail/userdb.db')dnl
define(`confPRIVACY_FLAGS', `authwarnings,novrfy,noexpn,restrictqrun')dnl
define(`confAUTH_OPTIONS', `A')dnl
dnl #
dnl # The following allows relaying if the user authenticates, and disallows
dnl # plaintext authentication (PLAIN/LOGIN) on non-TLS links
dnl #
dnl define(`confAUTH_OPTIONS', `A p')dnl
dnl #
dnl # PLAIN is the preferred plaintext authentication method and used by
dnl # Mozilla Mail and Evolution, though Outlook Express and other MUAs do
dnl # use LOGIN. Other mechanisms should be used if the connection is not
dnl # guaranteed secure.
dnl #
dnl TRUST_AUTH_MECH(`EXTERNAL DIGEST-MD5 CRAM-MD5 LOGIN PLAIN')dnl
dnl define(`confAUTH_MECHANISMS', `EXTERNAL GSSAPI DIGEST-MD5 CRAM-MD5 LOGIN PLAIN')dnl
dnl #
dnl # Rudimentary information on creating certificates for sendmail TLS:
dnl #     make -C /usr/share/ssl/certs usage
dnl #
dnl define(`confCACERT_PATH',`/usr/share/ssl/certs')
dnl define(`confCACERT',`/usr/share/ssl/certs/ca-bundle.crt')
dnl define(`confSERVER_CERT',`/usr/share/ssl/certs/sendmail.pem')
dnl define(`confSERVER_KEY',`/usr/share/ssl/certs/sendmail.pem')
dnl #
dnl # This allows sendmail to use a keyfile that is shared with OpenLDAP's
dnl # slapd, which requires the file to be readble by group ldap
dnl #
dnl define(`confDONT_BLAME_SENDMAIL',`groupreadablekeyfile')dnl
dnl #
dnl define(`confTO_QUEUEWARN', `4h')dnl
dnl define(`confTO_QUEUERETURN', `5d')dnl
dnl define(`confQUEUE_LA', `12')dnl
dnl define(`confREFUSE_LA', `18')dnl
define(`confTO_IDENT', `0')dnl
dnl FEATURE(delay_checks)dnl
FEATURE(`no_default_msa',`dnl')dnl
FEATURE(`smrsh',`/usr/sbin/smrsh')dnl
FEATURE(`mailertable',`hash -o /etc/mail/mailertable.db')dnl
FEATURE(`virtusertable',`hash -o /etc/mail/virtusertable.db')dnl
FEATURE(redirect)dnl
FEATURE(always_add_domain)dnl
FEATURE(use_cw_file)dnl
FEATURE(use_ct_file)dnl
dnl #
dnl # The -t option will retry delivery if e.g. the user runs over his quota.
dnl #
FEATURE(local_procmail,`',`procmail -t -Y -a $h -d $u')dnl
FEATURE(`access_db',`hash -T<TMPF> -o /etc/mail/access.db')dnl
FEATURE(`blacklist_recipients')dnl
EXPOSED_USER(`root')dnl
dnl #
dnl # The following causes sendmail to only listen on the IPv4 loopback address
dnl # 127.0.0.1 and not on any other network devices. Remove the loopback
dnl # address restriction to accept email from the internet or intranet.
dnl #
DAEMON_OPTIONS(`Port=smtp, Addr=127.0.0.1, Name=MTA')dnl
dnl #
dnl # The following causes sendmail to additionally listen to port 587 for
```

```
dnl # mail from MUAs that authenticate. Roaming users who can't reach their
dnl # preferred sendmail daemon due to port 25 being blocked or redirected find
dnl # this useful.
dnl #
dnl DAEMON_OPTIONS(`Port=submission, Name=MSA, M=Ea')dnl
dnl #
dnl # The following causes sendmail to additionally listen to port 465, but
dnl # starting immediately in TLS mode upon connecting. Port 25 or 587 followed
dnl # by STARTTLS is preferred, but roaming clients using Outlook Express can't
dnl # do STARTTLS on ports other than 25. Mozilla Mail can ONLY use STARTTLS
dnl # and doesn't support the deprecated smtps; Evolution <1.1.1 uses smtps
dnl # when SSL is enabled-- STARTTLS support is available in version 1.1.1.
dnl #
dnl # For this to work your OpenSSL certificates must be configured.
dnl #
dnl DAEMON_OPTIONS(`Port=smtps, Name=TLSMTA, M=s')dnl
dnl #
dnl # The following causes sendmail to additionally listen on the IPv6 loopback
dnl # device. Remove the loopback address restriction listen to the network.
dnl #
dnl # NOTE: binding both IPv4 and IPv6 daemon to the same port requires
dnl #       a kernel patch
dnl #
dnl DAEMON_OPTIONS(`port=smtp,Addr=::1, Name=MTA-v6, Family=inet6')dnl
dnl #
dnl # We strongly recommend not accepting unresolvable domains if you want to
dnl # protect yourself from spam. However, the laptop and users on computers
dnl # that do not have 24x7 DNS do need this.
dnl #
FEATURE(`accept_unresolvable_domains')dnl
dnl #
dnl FEATURE(`relay_based_on_MX')dnl
dnl #
dnl # Also accept email sent to "localhost.localdomain" as local email.
dnl #
LOCAL_DOMAIN(`localhost.localdomain')dnl
dnl #
dnl # The following example makes mail from this host and any additional
dnl # specified domains appear to be sent from mydomain.com
dnl #
MASQUERADE_AS(`ten0808080.com')dnl
dnl #
dnl # masquerade not just the headers, but the envelope as well
dnl #
FEATURE(masquerade_envelope)dnl
dnl #
dnl # masquerade not just @mydomainalias.com, but @*.mydomainalias.com as well
dnl #
dnl FEATURE(masquerade_entire_domain)dnl
dnl #
dnl MASQUERADE_DOMAIN(localhost)dnl
dnl MASQUERADE_DOMAIN(localhost.localdomain)dnl
dnl MASQUERADE_DOMAIN(mydomainalias.com)dnl
dnl MASQUERADE_DOMAIN(mydomain.lan)dnl
MAILER(smtp)dnl
MAILER(procmail)dnl
Cwlocalhost.localdomain
```

## Knockdaemon

## /usr/local/knockdaemon/knockdaemon

```
#!/usr/bin/perl

##############################################################
#
# $Id: knockdaemon,v 1.13 2003/03/03 07:32:11 martink Exp $
#
# Copyright 2003 Martin Krzywinski (martink@bcgsc.ca)
#
# This file is part of a Perl port knocking implementation.
#
# This port knocking implementation is free software; you can redistribute it and/or
modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation; either version 2 of the License, or
# (at your option) any later version.
#
# This port knocking implementation is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with Clusterpunch; if not, write to the Free Software
# Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
```

```
#
################################################################

################################################################
#
# http://mkweb.bcgsc.ca/portknocking
#
# WARNING
# This implementation is a PROTOTYPE and will always
# be a PROTOTYPE. The knockclient/knockdaemon pair of scripts
# is designed to illustrate the port knocking method. Don't use
# these scripts in a production environment. Don't expect a lot
# of functionality. If you want to write your own implementation,
# let me know and I'll post it on the site.
#
# OK
# Now you can read on ;)
#
################################################################

################################################################
#
# The knockdaemon utility continuously tails the firewall
# log file looking for lines containing the string DENY which
# also contain
#
# PROTO=TCP and SRC=IP.IP.IP.IP DST=IP.IP.IP.IP SPT=PORT DST=PORT
#
# The first IP address is the remote IP and the second PORT is
# the local port. The local port value is captured and
# inserted into a queue associated with the remote IP. Once
# the queue reaches length 8, we try to decrypt it, verify
# the checksum and act accordingly:
#
#  - time flag = 0   => open port
#  - time flag > 0   => close port
#
#
################################################################

use strict;
use File::Tail;
use Crypt::CBC;
#use Schedule::At;
use Math::VecStat qw(sum);
use POSIX qw(strftime);
use Pod::Usage;
use Getopt::Long;
use constant PORTMIN     => 745;
use constant KNOCKLENGTH  => 8;
use constant KEY          => "knock";
use constant CIPHER       => "Blowfish";
use vars qw($opt_file $opt_help $opt_man);

my $pod_usage_message = "PORT KNOCKING Prototype Perl Implementation\n\nknockdaemon -
server for monitoring firewall log file for incoming knocks.\n";

GetOptions("file=s","help","man") || pod2usage({-message=>$pod_usage_message});

pod2usage({-message=>$pod_usage_message}) if ($opt_help);
pod2usage(-verbose => 2) if ($opt_man);

die "cannot open firewall log file (-file FILENAME)" if ! -e $opt_file || ! -r _;

my $file = File::Tail->new(name=>"$opt_file",maxinterval=>2);

my %QUEUE;
while(defined(my $line=$file->read)) {
    # pay attention only to DENY packets
    next unless $line =~ /DENY/;
    next unless $line =~ /PROTO=TCP/;
```

```
    if($line =~ /SRC=([\d.]+).* DPT=(\d+)/) {
        my ($ip,$port) = ($1,$2);
        # pay attention only to ports allocated for knocking
        next if ($port < PORTMIN || $port >  PORTMIN+255);

        print "knock from $ip to port $port\n";

        # push this port to each queue item
        $QUEUE{$ip} = [] if ! $QUEUE{$ip};
        push(@{$QUEUE{$ip}},$port);
        print "current queue ",join(" ",@{$QUEUE{$ip}}),"\n";
        # if the queue is of the expected length, process it
        if(@{$QUEUE{$ip}} == KNOCKLENGTH) {
          ProcessQueue($QUEUE{$ip});
          print "current queue @{$QUEUE{$ip}}\n";
        }
      }
  }

sub ProcessQueue {
  my $queue = shift;
  # try to decrypt the queue contents
  my $cipher = Crypt::CBC->new({key        => KEY,
                                iv         =>"01234567",
                                prepend_iv => 0,
                                cipher     => CIPHER});
  my @data = unpack("C*",
                    $cipher->decrypt(pack("C*",
                                          map {$_ - PORTMIN } @$queue)));
  # decrypted list must have 7 elements, otherwise remove oldest item and keep listening
  if(@data != 7) {
    print "ERROR could not decrypt properly\n";
    shift(@$queue);
    return;
  }
  print "Got data ",join(" ",@data),"\n";
  # check crc of knock sequence
  if ((sum(@data[0..@data-2]) % 255) == $data[-1]) {
    # extract information from the decrypted list
    my ($remoteip,$localport,$time) = (join(".",@data[0..3]),$data[4],$data[5]);
    print "$remoteip $localport $time\n";
    # open port to remote ip
    system("/sbin/iptables -A ssh_rule -p tcp -s $remoteip/32 -i eth0 --dport $localport
-j ACCEPT") if $time != 255;
    # close the port if time > 0
    if($time > 0) {
      system("/sbin/iptables -F ssh_rule ");
    }
    # clear the queue
    @$queue = ();
  } else {
    print "ERROR bad crc\n";
    shift(@$queue);
  }
}

=pod

=head1 NAME

knockdaemon - port knocking server responsible for monitoring the firewall log file for
incoming knocks generated by F<knockclient> and adjusting the firewall rules accordingly.

=head1 SYNOPSIS

  > knockdaemon -f /var/log/firewall

Monitor the IPCHAINS-like firewall file F</var/log/firewall>.

  > knockdaemon -h
```

```
View usage help.

  > knockdaemon -man

View full manpage.

=head1 OPTIONS

=over

=item Parameters

Longer command line parameters are supported.

  > knockdaemon -file /var/log/firewall

=item Cipher

By default, the knock sequence will be decrypted using a Blowfish cipher. To change the
cipher alter the line

  use constant CIPHER => "Blowfish";

in both the knockclient and knockdaemon.

=item Passphrase

The passphrase "knock" is used by default to decrypt the knock sequence. To change the
passphrase alter the line

  use constant KEY => "knock";

in both the knockclient and knockdaemon.

=item Port Mapping

The encrypted knock values (0-255) are mapped onto ports 745-1000. To change the lower
range alter the line

  use constant PORTMIN => 745;

in both the knockclient and knockdaemon.

=item Initialization Vector

No initialization vector is used in the cipher. This results in the same encrypted knock
sequence for the same input, but shortens the encrypted knock sequence. Encrypting 7
unsigned chars (4 IP bytes, port byte, time byte and checksum byte) results in an
encrypted knock sequence of 8 unsigned chars.

=back

=head1 DESCRIPTION

The F<knockdaemon> is the server program and is part of the B<Port Knocking> Perl
prototype. The client side script is F<knockclient>. Refer to L<knockclient> for details
about this script.

The F<knockdaemon> is reponsible for monitoring the firewall log file, keeping track of
all port connection attempts to ports in the range PORTMIN - PORTMIN+255 from all IPs,
detecting the presence of properly formatted knock sequences and altering the firewall
rules accordingly.

=head2 Example

For example if port ssh/22 (B<-p 22>) were to be opened for 15 minutes (B<-t 15>) to
client 142.103.205.1 (B<-c 142.103.205.1>) on firewall at IPREMOTE the command would be

  knockclient -r IPREMOTE -c 142.103.205.1 -p 22 -t 15

The encrypted knock sequence is (see L<knockclient> for details)
```

```
  966 914 795 964 831 862 807 932

The F<knockdaemon>'s role is to detect this sequence, decrypt it, check that the checksum
is valid and adjust the firewall rule set. The firewall rules are added using

  /sbin/ipchains -I input -p tcp -s $IPCLIENT/32 -d 0/0 PORT -j ACCEPT

and deleted with

  /sbin/ipchains -D input -p tcp -s $IPCLIENT/32 -d 0/0 PORT -j DENY

where IPCLIENT (e.g. 142.103.205.1) is the client from which the connection to PORT is
expected.

=head1 AUTHOR

Martin Krzywinski, martink@bcgsc.ca

=head1 SEE ALSO

F<knockclient>

=head1 HISTORY

=over

=item 25 February 2002

Initial Perl prototype

=back

=head1 COPYRIGHT

(c) 2003 Martin Krzywinski

All rights reserved. This port knocking implementation is free software; you can
redistribute it and/or modify it under the terms of the GNU General Public License as
published by the Free Software Foundation; either version 2 of the License, or (at your
option) any later version.

Refer to COPYING in this distribution for the complete GPL license

=cut
```

## Tripwire policy

/etc/tripwire/twcfg.txt

```
@@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=SnortSensor;

@@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 that are changed infrequently but accessed often
SEC_LOG       = $(Growing) ;         # Files that grow, but that should never change ownership
SEC_INVARIANT = +tpug ;              # Directories that should never change permission or ownership
SIG_LOW       = 33 ;                 # Non-critical files that are of minimal security impact
SIG_MED       = 66 ;                 # Non-critical files that are of significant security impact
SIG_HI        = 100 ;                # Critical files that are significant points of vulnerability


# Tripwire Binaries
(
  rulename = "Tripwire Binaries",
  severity = $(SIG_HI)
)
{
```

```
  $(TWBIN)/siggen                       -> $(SEC_BIN) ;
  $(TWBIN)/tripwire                     -> $(SEC_BIN) ;
  $(TWBIN)/twadmin                      -> $(SEC_BIN) ;
  $(TWBIN)/twprint                      -> $(SEC_BIN) ;
}

# Tripwire Data Files - Configuration Files, Policy Files, Keys, Reports, Databases
(
  rulename = "Tripwire Data Files",
  severity = $(SIG_HI)
)
{
  # NOTE: We remove the inode attribute because when Tripwire creates a backup,
  # it does so by renaming the old file and creating a new one (which will
  # have a new inode number).  Inode is left turned on for keys, which shouldn't
  # ever change.

  # NOTE: The first integrity check triggers this rule and each integrity check
  # afterward triggers this rule until a database update is run, since the
  # database file does not exist before that point.

  $(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) ;

  #don't scan the individual reports
  $(TWREPORT)                           -> $(SEC_CONFIG) (recurse=0) ;
}




# Commonly accessed directories that should remain static with regards to owner and group
(
  rulename = "Invariant Directories",
  severity = $(SIG_MED)
)
{
  /                                     -> $(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 Administraton Programs",
  severity = $(SIG_HI)
)
{
  /sbin/badblocks                       -> $(SEC_CRIT) ;
  /sbin/convertquota                    -> $(SEC_CRIT) ;
  /sbin/debugfs                         -> $(SEC_CRIT) ;
  /sbin/dumpe2fs                        -> $(SEC_CRIT) ;
  /sbin/e2fsck                          -> $(SEC_CRIT) ;
  /sbin/e2label                         -> $(SEC_CRIT) ;
  /sbin/fdisk                           -> $(SEC_CRIT) ;
  /sbin/fsck                            -> $(SEC_CRIT) ;
  /sbin/fsck.ext2                       -> $(SEC_CRIT) ;
  /sbin/fsck.ext3                       -> $(SEC_CRIT) ;
  /sbin/hdparm                          -> $(SEC_CRIT) ;
  /sbin/mke2fs                          -> $(SEC_CRIT) ;
  /sbin/mkfs                            -> $(SEC_CRIT) ;
  /sbin/mkfs.ext2                       -> $(SEC_CRIT) ;
  /sbin/mkinitrd                        -> $(SEC_CRIT) ;
  /sbin/mkraid                          -> $(SEC_CRIT) ;
  /sbin/mkswap                          -> $(SEC_CRIT) ;
  /sbin/pam_console_apply               -> $(SEC_CRIT) ;
  /sbin/quotacheck                      -> $(SEC_CRIT) ;
  /sbin/quotaon                         -> $(SEC_CRIT) ;
  /sbin/raidstart                       -> $(SEC_CRIT) ;
  /sbin/resize2fs                       -> $(SEC_CRIT) ;
  /sbin/sfdisk                          -> $(SEC_CRIT) ;
  /sbin/tune2fs                         -> $(SEC_CRIT) ;
  /bin/chgrp                            -> $(SEC_CRIT) ;
  /bin/chmod                            -> $(SEC_CRIT) ;
  /bin/chown                            -> $(SEC_CRIT) ;
  /bin/cp                               -> $(SEC_CRIT) ;
  /bin/cpio                             -> $(SEC_CRIT) ;
  /bin/mount                            -> $(SEC_CRIT) ;
  /bin/umount                           -> $(SEC_CRIT) ;
  /bin/mkdir                            -> $(SEC_CRIT) ;
  /bin/mknod                            -> $(SEC_CRIT) ;
  /bin/mktemp                           -> $(SEC_CRIT) ;
  /bin/rm                               -> $(SEC_CRIT) ;
  /bin/rmdir                            -> $(SEC_CRIT) ;
  /bin/touch                            -> $(SEC_CRIT) ;
```

```
}

  ###################################
 #                                 ##
################################### #
#                                 # #
# Kernel Administration Programs # #
#                                ##
###################################

(
  rulename = "Kernel Administration Programs",
  severity = $(SIG_HI)
)
{
  /sbin/ctrlaltdel                    -> $(SEC_CRIT) ;
  /sbin/depmod                        -> $(SEC_CRIT) ;
  /sbin/insmod                        -> $(SEC_CRIT) ;
  /sbin/insmod.static                 -> $(SEC_CRIT) ;
  /sbin/insmod_ksymoops_clean         -> $(SEC_CRIT) ;
  /sbin/klogd                         -> $(SEC_CRIT) ;
  /sbin/ldconfig                      -> $(SEC_CRIT) ;
  /sbin/minilogd                      -> $(SEC_CRIT) ;
  /sbin/modinfo                       -> $(SEC_CRIT) ;
  /sbin/pivot_root                    -> $(SEC_CRIT) ;
  /sbin/sysctl                        -> $(SEC_CRIT) ;
}

  ######################
 #                    ##
###################### #
#                   # #
# Networking Programs # #
#                  ##
######################

(
  rulename = "Networking Programs",
  severity = $(SIG_HI)
)
{
  /etc/sysconfig/network-scripts/ifdown                -> $(SEC_CRIT) ;
  /etc/sysconfig/network-scripts/ifdown-ippp           -> $(SEC_CRIT) ;
  /etc/sysconfig/network-scripts/ifdown-ipv6           -> $(SEC_CRIT) ;
  /etc/sysconfig/network-scripts/ifdown-isdn           -> $(SEC_CRIT) ;
  /etc/sysconfig/network-scripts/ifdown-post           -> $(SEC_CRIT) ;
  /etc/sysconfig/network-scripts/ifdown-ppp            -> $(SEC_CRIT) ;
  /etc/sysconfig/network-scripts/ifdown-sit            -> $(SEC_CRIT) ;
  /etc/sysconfig/network-scripts/ifdown-sl             -> $(SEC_CRIT) ;
  /etc/sysconfig/network-scripts/ifup                  -> $(SEC_CRIT) ;
  /etc/sysconfig/network-scripts/ifup-aliases          -> $(SEC_CRIT) ;
  /etc/sysconfig/network-scripts/ifup-ippp             -> $(SEC_CRIT) ;
  /etc/sysconfig/network-scripts/ifup-ipv6             -> $(SEC_CRIT) ;
  /etc/sysconfig/network-scripts/ifup-isdn             -> $(SEC_CRIT) ;
  /etc/sysconfig/network-scripts/ifup-plip             -> $(SEC_CRIT) ;
  /etc/sysconfig/network-scripts/ifup-plusb            -> $(SEC_CRIT) ;
  /etc/sysconfig/network-scripts/ifup-post             -> $(SEC_CRIT) ;
  /etc/sysconfig/network-scripts/ifup-ppp              -> $(SEC_CRIT) ;
  /etc/sysconfig/network-scripts/ifup-routes           -> $(SEC_CRIT) ;
  /etc/sysconfig/network-scripts/ifup-sit              -> $(SEC_CRIT) ;
  /etc/sysconfig/network-scripts/ifup-sl               -> $(SEC_CRIT) ;
  /etc/sysconfig/network-scripts/ifup-wireless         -> $(SEC_CRIT) ;
  /etc/sysconfig/network-scripts/network-functions     -> $(SEC_CRIT) ;
  /etc/sysconfig/network-scripts/network-functions-ipv6  -> $(SEC_CRIT) ;
  /bin/ping                           -> $(SEC_CRIT) ;
  /sbin/agetty                        -> $(SEC_CRIT) ;
  /sbin/arp                           -> $(SEC_CRIT) ;
  /sbin/arping                        -> $(SEC_CRIT) ;
  /sbin/ether-wake                    -> $(SEC_CRIT) ;
  /sbin/ifcfg                         -> $(SEC_CRIT) ;
  /sbin/ifconfig                      -> $(SEC_CRIT) ;
  /sbin/ifdown                        -> $(SEC_CRIT) ;
  /sbin/ifenslave                     -> $(SEC_CRIT) ;
  /sbin/ifup                          -> $(SEC_CRIT) ;
  /sbin/ip                            -> $(SEC_CRIT) ;
  /sbin/ipmaddr                       -> $(SEC_CRIT) ;
  /sbin/iptables                      -> $(SEC_CRIT) ;
  /sbin/iptables-restore              -> $(SEC_CRIT) ;
  /sbin/iptables-save                 -> $(SEC_CRIT) ;
  /sbin/iptunnel                      -> $(SEC_CRIT) ;
  /sbin/mingetty                      -> $(SEC_CRIT) ;
  /sbin/nameif                        -> $(SEC_CRIT) ;
  /sbin/netreport                     -> $(SEC_CRIT) ;
  /sbin/plipconfig                    -> $(SEC_CRIT) ;
  /sbin/ppp-watch                     -> $(SEC_CRIT) ;
  /sbin/route                         -> $(SEC_CRIT) ;
  /sbin/slattach                      -> $(SEC_CRIT) ;
  /sbin/tc                            -> $(SEC_CRIT) ;
}

  ###################################
 #                                 ##
################################### #
#                                 # #
# System Administration Programs # #
```

```
#                               ##
#################################

(
  rulename = "System Administration Programs",
  severity = $(SIG_HI)
)
{
  /sbin/chkconfig                       -> $(SEC_CRIT) ;
  /sbin/fuser                           -> $(SEC_CRIT) ;
  /sbin/halt                            -> $(SEC_CRIT) ;
  /sbin/init                            -> $(SEC_CRIT) ;
  /sbin/initlog                         -> $(SEC_CRIT) ;
  /sbin/install-info                    -> $(SEC_CRIT) ;
  /sbin/killall5                        -> $(SEC_CRIT) ;
  /sbin/pam_tally                       -> $(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)
)
{
  /bin/setserial                        -> $(SEC_CRIT) ;
  /sbin/blockdev                        -> $(SEC_CRIT) ;
  /sbin/elvtune                         -> $(SEC_CRIT) ;
  /sbin/hwclock                         -> $(SEC_CRIT) ;
  /sbin/losetup                         -> $(SEC_CRIT) ;
  /sbin/mii-tool                        -> $(SEC_CRIT) ;
}

  ##############################
 #                            ##
############################## #
#                            # #
# System Information Programs # #
#                            ##
##############################
(
  rulename = "System Information Programs",
  severity = $(SIG_HI)
)
{
  /sbin/consoletype                     -> $(SEC_CRIT) ;
  /sbin/kernelversion                   -> $(SEC_CRIT) ;
  /sbin/runlevel                        -> $(SEC_CRIT) ;
}

  ##################################
 #                                ##
################################## #
#                                # #
# Application Information Programs # #
#                                ##
##################################

(
  rulename = "Application Information Programs",
  severity = $(SIG_HI)
)
{
  /sbin/genksyms                        -> $(SEC_CRIT) ;
  /sbin/rtmon                           -> $(SEC_CRIT) ;
}

  ########################
 #                      ##
######################## #
#                      # #
# Shell Related Programs # #
#                      ##
########################
(
  rulename = "Shell Related Programs",
  severity = $(SIG_HI)
)
{
```

```
  /sbin/getkey                          -> $(SEC_CRIT) ;
  /sbin/nash                            -> $(SEC_CRIT) ;
}


  ################
 #              ##
############### #
#             # #
# OS Utilities # #
#             ##
###############
(
  rulename = "Operating System Utilities",
  severity = $(SIG_HI)
)
{
  /bin/arch                             -> $(SEC_CRIT) ;
  /bin/basename                         -> $(SEC_CRIT) ;
  /bin/cat                              -> $(SEC_CRIT) ;
  /bin/cut                              -> $(SEC_CRIT) ;
  /bin/date                             -> $(SEC_CRIT) ;
  /bin/dd                               -> $(SEC_CRIT) ;
  /bin/df                               -> $(SEC_CRIT) ;
  /bin/dmesg                            -> $(SEC_CRIT) ;
  /bin/doexec                           -> $(SEC_CRIT) ;
  /bin/echo                             -> $(SEC_CRIT) ;
  /bin/ed                               -> $(SEC_CRIT) ;
  /bin/egrep                            -> $(SEC_CRIT) ;
  /bin/false                            -> $(SEC_CRIT) ;
  /bin/fgrep                            -> $(SEC_CRIT) ;
  /bin/gawk                             -> $(SEC_CRIT) ;
  /bin/grep                             -> $(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/pgawk                            -> $(SEC_CRIT) ;
  /bin/ps                               -> $(SEC_CRIT) ;
  /bin/rpm                              -> $(SEC_CRIT) ;
  /bin/sed                              -> $(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/true                             -> $(SEC_CRIT) ;
  /bin/usleep                           -> $(SEC_CRIT) ;
  /bin/vi                               -> $(SEC_CRIT) ;
  /bin/zcat                             -> $(SEC_CRIT) ;
  /sbin/sln                             -> $(SEC_CRIT) ;
}

  ############################
 #                          ##
############################ #
#                          # #
# Critical Utility Sym-Links # #
#                          ##
############################
(
  rulename = "Critical Utility Sym-Links",
  severity = $(SIG_HI)
)
{
  /sbin/clock                           -> $(SEC_CRIT) ;
  /sbin/kallsyms                        -> $(SEC_CRIT) ;
  /sbin/ksyms                           -> $(SEC_CRIT) ;
  /sbin/lsmod                           -> $(SEC_CRIT) ;
  /sbin/modprobe                        -> $(SEC_CRIT) ;
  /sbin/pidof                           -> $(SEC_CRIT) ;
  /sbin/poweroff                        -> $(SEC_CRIT) ;
  /sbin/quotaoff                        -> $(SEC_CRIT) ;
  /sbin/raid0run                        -> $(SEC_CRIT) ;
  /sbin/raidhotadd                      -> $(SEC_CRIT) ;
  /sbin/raidhotremove                   -> $(SEC_CRIT) ;
  /sbin/raidstop                        -> $(SEC_CRIT) ;
  /sbin/reboot                          -> $(SEC_CRIT) ;
  /sbin/rmmod                           -> $(SEC_CRIT) ;
  /sbin/swapoff                         -> $(SEC_CRIT) ;
  /sbin/telinit                         -> $(SEC_CRIT) ;
  /bin/awk                              -> $(SEC_CRIT) ;
  /bin/bash2                            -> $(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)
)
{
  /usr/tmp                              -> $(SEC_INVARIANT) ;
  /var/tmp                              -> $(SEC_INVARIANT) ;
  /tmp                                  -> $(SEC_INVARIANT) ;
}

  ###############
 #             ##
############### #
#             # #
# Local files # #
#              ##
###############
(
  rulename = "User binaries",
  severity = $(SIG_MED)
)
{
  /sbin                                 -> $(SEC_BIN) (recurse = 1) ;
  /usr/bin                              -> $(SEC_BIN) (recurse = 1) ;
  /usr/sbin                             -> $(SEC_BIN) (recurse = 1) ;
  /usr/local/bin                        -> $(SEC_BIN) (recurse = 1) ;
}

(
  rulename = "Shell Binaries",
  severity = $(SIG_HI)
)
{
  /bin/bash                             -> $(SEC_BIN) ;
  /bin/sh                               -> $(SEC_BIN) ;
  /sbin/nologin                         -> $(SEC_BIN) ;
}

(
  rulename = "Security Control",
  severity = $(SIG_HI)
)
{
  /etc/group                            -> $(SEC_CRIT) ;
  /etc/security                         -> $(SEC_CRIT) ;
  #/var/spool/cron/crontabs              -> $(SEC_CRIT) ; # Uncomment when this file exists
}

(
  rulename = "Boot Scripts",
  severity = $(SIG_HI)
)
{
  /etc/rc.fire                               -> $(SEC_CONFIG) ;
}

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

# Libraries
(
  rulename = "Libraries",
  severity = $(SIG_MED)
)
{
```

```
  /usr/lib                              -> $(SEC_BIN) ;
  /usr/local/lib                        -> $(SEC_BIN) ;
}


  ##################################################
 #                                                ##
################################################## #
#                                                # #
# Critical System Boot Files                       # #
# These files are critical to a correct system boot. # #
#                                                ##
##################################################

(
  rulename = "Critical system boot files",
  severity = $(SIG_HI)
)
{
     /boot                              -> $(SEC_CRIT) ;
     /sbin/grub                         -> $(SEC_CRIT) ;
     /sbin/grub-install                 -> $(SEC_CRIT) ;
     /sbin/grub-md5-crypt               -> $(SEC_CRIT) ;
     /sbin/installkernel                -> $(SEC_CRIT) ;
     /sbin/lilo                         -> $(SEC_CRIT) ;
     /sbin/mkkerneldoth                 -> $(SEC_CRIT) ;
     !/boot/System.map ;
     !/boot/module-info ;
     /usr/share/grub/i386-redhat/e2fs_stage1_5      -> $(SEC_CRIT) ;
     /usr/share/grub/i386-redhat/fat_stage1_5       -> $(SEC_CRIT) ;
     /usr/share/grub/i386-redhat/ffs_stage1_5       -> $(SEC_CRIT) ;
     /usr/share/grub/i386-redhat/minix_stage1_5     -> $(SEC_CRIT) ;
     /usr/share/grub/i386-redhat/reiserfs_stage1_5  -> $(SEC_CRIT) ;
     /usr/share/grub/i386-redhat/stage1             -> $(SEC_CRIT) ;
     /usr/share/grub/i386-redhat/stage2             -> $(SEC_CRIT) ;
     /usr/share/grub/i386-redhat/vstafs_stage1_5    -> $(SEC_CRIT) ;
}
   ################################################
  ################################################
  # These files change every time the system boots ##
  ################################################
(
  rulename = "System boot changes",
  severity = $(SIG_HI)
)
{
     !/var/run/ftp.pids-all ; # Comes and goes on reboot.
     !/root/.enlightenment ;
     /dev/log                           -> $(SEC_CONFIG) ;
     /dev/cua0                          -> $(SEC_CONFIG) ;
     /dev/console                       -> $(SEC_CONFIG) -u ; # User ID may change on console login/logout.
     /dev/tty1                          -> $(SEC_CONFIG) ; # tty devices
     /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/apmd              -> $(SEC_CONFIG) ;
     /var/lock/subsys/keytable          -> $(SEC_CONFIG) ;
     /var/lock/subsys/network           -> $(SEC_CONFIG) ;
     /var/lock/subsys/ntpd              -> $(SEC_CONFIG) ;
     /var/lock/subsys/random            -> $(SEC_CONFIG) ;
     /var/lock/subsys/sendmail          -> $(SEC_CONFIG) ;
     /var/lock/subsys/sshd              -> $(SEC_CONFIG) ;
     /var/lock/subsys/syslog            -> $(SEC_CONFIG) ;
     /var/run                           -> $(SEC_CONFIG) ;
     /var/log                           -> $(SEC_CONFIG) ;
     /etc/ioctl.save                    -> $(SEC_CONFIG) ;
     /etc/issue.net                     -> $(SEC_CONFIG) -i ; # Inode number changes
     /etc/issue                         -> $(SEC_CONFIG) ;
     /etc/mtab                          -> $(SEC_CONFIG) -i ; # Inode number changes on any mount/unmount
     /lib/modules                       -> $(SEC_CONFIG) ;
     /etc/.pwd.lock                     -> $(SEC_CONFIG) ;
}

# These files change the behavior of the root account
(
  rulename = "Root config files",
  severity = 100
)
{
     /root                              -> $(SEC_CRIT) ; # Catch all additions to /root
     /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)
)
{
     /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 should be 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/modules.conf                 -> $(SEC_BIN) ;
     /etc/motd                         -> $(SEC_BIN) ;
     /etc/passwd                       -> $(SEC_CONFIG) ;
     /etc/passwd-                      -> $(SEC_CONFIG) ;
     /etc/profile.d                    -> $(SEC_BIN) ;
     /etc/rpc                          -> $(SEC_BIN) ;
     /etc/ssh                           -> $(SEC_BIN) ;
     /etc/sysconfig                    -> $(SEC_BIN) ;
     /etc/nsswitch.conf                -> $(SEC_BIN) ;
     /etc/yp.conf                      -> $(SEC_BIN) ;
     /etc/hosts                        -> $(SEC_CONFIG) ;
     /etc/inittab                      -> $(SEC_CONFIG) ;
     /etc/resolv.conf                  -> $(SEC_CONFIG) ;
     /etc/syslog.conf                  -> $(SEC_CONFIG) ;
}

  ####################
 #                  ##
################### #
#                 # #
# Critical devices # #
#                 ##
###################
(
  rulename = "Critical devices",
  severity = $(SIG_HI),
  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/driver/rtc                  -> $(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/mdstat                      -> $(Device) ;
     /proc/meminfo                     -> $(Device) ;
     /proc/cmdline                     -> $(Device) ;
     /proc/misc                        -> $(Device) ;
}

# Rest of critical system binaries
(
  rulename = "OS executables and libraries",
  severity = $(SIG_HI)
)
{
     /bin                              -> $(SEC_BIN) ;
     /lib                              -> $(SEC_BIN) ;
}
```

```
# Snort sensor specific files
(
  rulename = "Snort sensor specific files",
  severity = $(SIG_HI)
)
{
     /usr/local/knockdaemon             -> $(SEC_CRIT) ;
     /var/snort/sbin/                           -> $(SEC_CRIT) ;
     /etc/notify/                       -> $(SEC_CRIT) ;
}
```

```
#============================================================================
#
# 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
```

Last Updated: April 26th, 2018

# Upcoming Training


| | | | |
|---|---|---|---|
| **SANSFIRE 2018** | **Washington, DC** | **Jul 14, 2018 - Jul 21, 2018** | **Live Event** |
| **SANS Network Security 2018** | **Las Vegas, NV** | **Sep 23, 2018 - Sep 30, 2018** | **Live Event** |
| **SANS London October 2018** | **London, United Kingdom** | **Oct 15, 2018 - Oct 20, 2018** | **Live Event** |
| **SANS OnDemand** | **Online** | **Anytime** | **Self Paced** |
| **SANS SelfStudy** | **Books & MP3s Only** | **Anytime** | **Self Paced** |