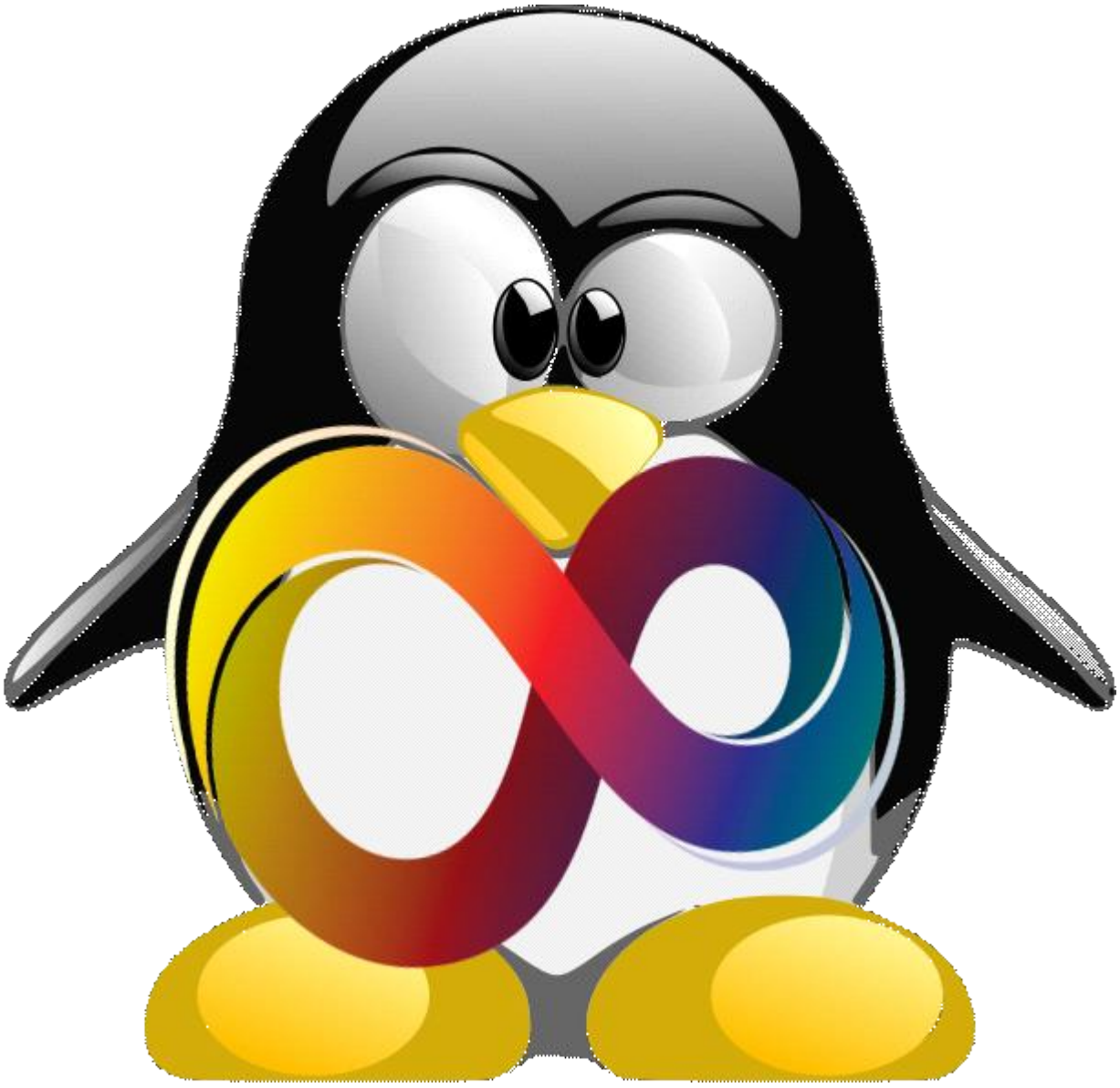
Chapter 1. INTRODUCTION

Welcome

to **Linux Checklist UNLIMITED**



Important Notes

1. ****DO THE FORENSICS QUESTION.****

2. **If you have done the forensics question, then read the competition scenario** before doing anything else. **If** there is a question, **answer the question** before proceeding to the next steps.

3. **If** you have done the forensics question **and if** you have read the competition scenario, **then glance over** the tools on the Tool List.

4. Consider setting up a strategy with the General Checklist. The checklist is sorted into color coordinated categories: Blue are the "very important" issues that should immediately be addressed; Yellow deals with hardening, requiring careful focus and no typos; Green deals with services and specific vulnerability patches; Red deals with search and destroy; and Gray focuses on more elaborate methods of system hardening that should only be addressed last. One strategy might be to initiate some Red scans in tty1 while you work on the Blue issues in tty2. Another strategy is to simply follow the checklist in order.

5. RTM (read the manual). Syntax: **man** *command_name*
   For example: **man ls**

   Also refer to search engines to answer questions, investigate tutorials, and learn how to further secure the system. Consider researching "Ubuntu / Debian security guides" and tutorials, searching for topics that you may not have already addressed.

Endlessly revert to step #4. Use Google dorks to your advantage.

Changelog

- Added hyperlinks to specific checklist page. (You can now click an option on the checklist, and it will take you straight to the command to enter.)
- Added UFW commands to Firewall page.
- Added SYN cookie protection to Firewall page.
- Added default inbound/outbound rules to Firewall page.
- Updated opasswd instructions on Security Policies page.
- Updated chage instructions on Security Policies page.
- Re-organized sections of Security Policies page.
- Added minimum password length information to Security Policies page.
- Updated password expiration information on Security Policies page.
- Updated specific checklist: Install and enable cracklib.
- Updated specific checklist: Set a minimum password length of 8.
- Updated specific checklist: Enable password history.
- Updated specific checklist: Enable password expiration.
- Updated specific checklist: Enable auditing.
- Updated specific checklist: Disable IPv6 if not needed.
- Updated specific checklist: Restrict DMESG.
- Updated specific checklist: Harden Linux Kernel parameters.
- Renamed page "Enabled Mandatory Access Control" to "Mandatory Access Control".
- Updated specific checklist: Enable apparmor.
- Updated specific checklist: Remove dangerous files from /var/www.
- Added upgrade distribution information to Updates page.
- Updated specific checklist: Check cronjobs.
- Updated specific checklist: Check startup directories for malicious files.
- Updated specific checklist: Find files with unwanted permissions.

- Updated specific checklist: Remove unnecessary media files.
- Updated specific checklist: Search for netcat and nc and remove them.
- Updated information on Search & Destroy page, View processes listening on ports section.
- Added Port Scanning and Vulnerability Scanning page.
- Updated information on Security Policies page.
- Updated specific checklist: enforce account lockout policy.
- Updated information on Search & Destroy page: secure ports.
- Updated specific checklist: Secure ports.
- Updated specific checklist: Clean DNS resolver.
- Updated specific checklist: Secure home directories.
- Updated specific checklist: Clear the issue banner.
- Updated specific checklist: Do the forensics questions.
- Updated general checklist: Forensics questions.

Chapter 1.1
ABOUT HACKING AND LINUX

Philosophy of Linux

Linux is a **FREE** operating system. Not necessarilly free in terms of price, but rather the **free**dom to run, copy, distribute, study, change, and improve the software.

A program can be classified as FREE if it abides to GNU's four essential freedoms:

0. The freedom to run the program as you wish, for any purpose.
1. The freedom to study how the program works, and the freedom to modify the program. Therefore, access to the source code is a necessary precondition.
2. The freedom to redistribute copies.
3. The freedom to distribute copies of your modified version of the software.

For a detailed explanation of the four freedoms, see:
https://www.gnu.org/philosophy/free-sw.html

Philosophy of Hackers

There exist multiple philosophies of hacking, but some of the most influential come from:

- **Richard Stallman** -- Founder of the Free Software Foundation and launched the GNU project.
- **Paul Graham** -- Major contributor to Lisp programming language, co-founder of Yahoo! Store, and essayist.
- **The Mentor** -- Author of "The Hacker Manifesto," a cornerstone of hacker culture.
- **Eric Steven Raymond** -- Open source advocate and author of "Rootless Root: The Unix Koans of Master Foo."

Here are excerpts from each philosopher of hacking:

Richard Stallman -- On Hacking

In June 2000, while visiting Korea, I did a fun hack that clearly illustrates the original and true meaning of the word "hacker".

I went to lunch with some GNU fans, and was sitting down to eat some tteokpaekki (*), when a waitress set down six chopsticks right in front of me. It occurred to me that perhaps these were meant for three people, but it was more amusing to imagine that I was supposed to use all six. I did not know any way to do that, so I realized that if I could come up with a way, it would be a hack. I started thinking. After a few seconds I had an idea.

First I used my left hand to put three chopsticks into my right hand. That was not so hard, though I had to figure out where to put them so that I could control them individually. Then I used my right hand to put the other three chopsticks into my left hand. That was hard, since I had to keep the three chopsticks already in my right hand from falling out. After a couple of tries I got it done.

Then I had to figure out how to use the six chopsticks. That was harder. I did not manage well with the left hand, but I succeeded in manipulating all three in the right hand. After a couple of minutes of practice and adjustment, I managed to pick up a piece of food using three sticks converging on it from three different directions, and put it in my mouth.

It didn't become easy—for practical purposes, using two chopsticks is completely superior. But precisely because using three in one hand is hard and ordinarily never thought of, it has "hack value", as my lunch companions immediately recognized. Playfully doing something difficult, whether useful or not, that is hacking.

I later told the Korea story to a friend in Boston, who proceded to put four chopsticks in one hand and use them as two pairs—picking up two different pieces of food at once, one with each pair. He had topped my hack. Was his action, too, a hack? I think so. Is he therefore a hacker? That depends on how much he likes to hack.

The hacking community developed at MIT and some other universities in the 1960s and 1970s. Hacking included a wide range of activities, from writing software, to practical jokes, to exploring the roofs and tunnels of the MIT campus. Other activities, performed far from MIT and far from computers, also fit hackers' idea of what hacking means: for instance, I think the controversial 1950s "musical piece" by John Cage, 4'33" (****), is more of a hack than a musical composition. The palindromic three-part piece written by Guillaume de Machaut in the 1300s, "Ma Fin Est Mon Commencement", was also a good hack, even better because it also sounds good as music. Puck appreciated hack value.

Read more: *https://stallman.org/articles/on-hacking.html*

Paul Graham -- The Word "Hacker"

To the popular press, "hacker" means someone who breaks into computers. Among programmers it means a good programmer. But the two meanings are connected. To programmers, "hacker" connotes mastery in the most literal sense: someone who can make a computer do what he wants—whether the computer wants to or not.

To add to the confusion, the noun "hack" also has two senses. It can be either a compliment or an insult. It's called a hack when you do something in an ugly way. But when you do something so clever that you somehow beat the system, that's also called a hack. The word is used more often in the former than the latter sense, probably because ugly solutions are more common than brilliant ones.

Believe it or not, the two senses of "hack" are also connected. Ugly and imaginative solutions have something in common: they both break the rules. And there is a gradual continuum between rule breaking that's merely ugly (using duct tape to attach something to your bike) and rule breaking that is brilliantly imaginative (discarding Euclidean space).

Hacking predates computers. When he was working on the Manhattan Project, Richard Feynman used to amuse himself by breaking into safes containing

secret documents. This tradition continues today. When we were in grad school, a hacker friend of mine who spent too much time around MIT had his own lock picking kit. (He now runs a hedge fund, a not unrelated enterprise.)

It is sometimes hard to explain to authorities why one would want to do such things. Another friend of mine once got in trouble with the government for breaking into computers. This had only recently been declared a crime, and the FBI found that their usual investigative technique didn't work. Police investigation apparently begins with a motive. The usual motives are few: drugs, money, sex, revenge. Intellectual curiosity was not one of the motives on the FBI's list. Indeed, the whole concept seemed foreign to them.

Those in authority tend to be annoyed by hackers' general attitude of disobedience. But that disobedience is a byproduct of the qualities that make them good programmers. They may laugh at the CEO when he talks in generic corporate newspeech, but they also laugh at someone who tells them a certain problem can't be solved. Suppress one, and you suppress the other.

Read more: *http://paulgraham.com/gba.html*

The Mentor -- The Conscience of a Hacker

Another one got caught today, it's all over the papers.
"Teenager Arrested in Computer Crime Scandal", "Hacker Arrested after
Bank Tampering"...

Damn kids. They're all alike.

But did you, in your three-piece psychology and 1950's
technobrain ever take a look behind the eyes of the Hacker? Did you ever
wonder what made him tick, what forces shaped him, what may have molded
him?

I am a Hacker, enter my world....

Mine is a world that begins with school... I'm smarter than most
of the other kids, this crap they teach us bores me...

Damn underachiever. They're all alike.

I made a discovery today. I found a computer. Wait a second,
this is cool. It does what I want it to. If it makes a mistake it's
because I screwed it up. Not because it doesn't like me...
Or feels threatened by me...

Read more: *http://www.usc.edu/~douglast/202/lecture23/manifesto.html*

Eric Steven Raymond -- Master Foo and the Ten Thousand Lines

Master Foo once said to a visiting programmer: "There is more Unix-nature in one line of shell script than there is in ten thousand lines of C."

The programmer, who was very proud of his mastery of C, said: "How can this be? C is the language in which the very kernel of Unix is implemented!"

Master Foo replied: "That is so. Nevertheless, there is more Unix-nature in one line of shell script than there is in ten thousand lines of C."

The programmer grew distressed. "But through the C language we experience the enlightenment of the Patriarch Ritchie! We become as one with the operating system and the machine, reaping matchless performance!"

Master Foo replied: "All that you say is true. But there is still more Unix-nature in one line of shell script than there is in ten thousand lines of C."

The programmer scoffed at Master Foo and rose to depart. But Master Foo nodded to his student Nubi, who wrote a line of shell script on a nearby whiteboard, and said: "Master programmer, consider this pipeline. Implemented in pure C, would it not span ten thousand lines?"

The programmer muttered through his beard, contemplating what Nubi had written. Finally he agreed that it was so.

"And how many hours would you require to implement and debug that C program?" asked Nubi.

"Many," admitted the visiting programmer. "But only a fool would spend the time to do that when so many more worthy tasks await him."

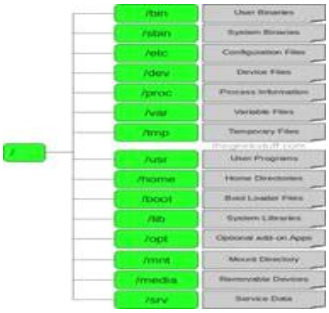
"And who better understands the Unix-nature?" Master Foo asked. "Is it he who writes the ten thousand lines, or he who, perceiving the emptiness of the task, gains merit by not coding?"

Upon hearing this, the programmer was enlightened.

Read more: *http://www.catb.org/~esr/writings/unix-koans/*

Chapter 1.2
GETTING STARTED WITH LINUX

Paths, Terminology, /etc



Reboot: **sudo reboot**
Shutdown: **sudo shutdown -h now**

SOURCE: http://rute.2038bug.com/node7.html.gz
SOURCE: http://www.thegeekstuff.com/2010/09/linux-file-system-structure/

/root -- root user's home directory
/bin -- contains binary executables
/sbin -- contains **system** binary executables
/etc -- contains config files required for all programs
/dev -- contains device files, like USB drives and sound cards
/proc -- contains information about system processes
/var -- contains variable files that are expected to grow (databases, logs, etc.)
/usr -- contains binaries, source code, and documentation for second-level programs
/home -- contains user directories where personal files go (My Documents)
/boot -- contains file required to boot the system
/lib -- contains library files supported by binaries under **/bin** and **/sbin**
/opt -- contains optional add-on applications
/mnt -- a temporary mount directory where admins can mount devices, file systems, etc.
/media -- contains removable media devices (CD-ROMs, floppy, etc.)
/srv -- contains server-specific related data

Some Basic Commands

bc

A calculator program that handles arbitrary precision (very large) numbers. It is useful for doing any kind of calculation on the command-line. Its use is left as an exercise.

cal [[0-12] 1-9999]

Prints out a nicely formatted calender of the current month, a specified month, or a specified whole year. Try cal 1 for fun, and cal 9 1752,
when the pope had a few days scrapped to compensate for round-off error.

cat <filename> [<filename> ...]

Writes the contents of all the files listed to the screen. cat can join a lot of files together with cat <filename> <filename> ... > <newfile>. The
file <newfile> will be an end-on-end *concatenation* of all the files specified.

clear

Erases all the text in the current terminal.

date

Prints out the current date and time. (The command time, though, does something entirely different.)

df

Stands for *disk free* and tells you how much free space is left on your system. The available space usually has the units of kilobytes (1024 bytes) (although on some other UNIX systems this will be 512 bytes or 2048 bytes). The right-most column tells the directory (in combination with any directories below that) under which that much space is available.

dircmp

Directory compare. This command compares directories to see if changes have been made between them. You will often want to see where two trees differ (e.g., check for missing files), possibly on different computers. Run man dircmp (that is, dircmp(1)). (This is a System 5 command and is not present on LINUX. You can, however, compare directories with the Midnight Commander, mc).

du <directory>

Stands for *disk usage* and prints out the amount of space occupied by a directory. It recurses into any subdirectories and can print only a summary with du -s <directory>. Also try du --max-depth=1 /var and du -x / on a system with /usr and /home on separate *partitions*.

dmesg

Prints a complete log of all messages printed to the screen during the bootup process. This is useful if you blinked when your machine was
initializing. These messages might not yet be meaningful, however.

echo

Prints a message to the terminal. Try echo 'hello there', echo $[10*3+2], echo `$[10*3+2]'. The command echo -e allows interpretation of certain *backslash* sequences, for example echo -e "\a", which prints a *bell*, or in other words, beeps the terminal. echo -n does the same without printing the trailing newline. In other words, it does not cause a wrap to the next line after the text is printed. echo -e -n "\b", prints a back-space character only, which will erase the last character printed.

exit

Logs you out.

expr <expression>

Calculates the numerical expression expression. Most arithmetic operations that you are accustomed to will work. Try expr 5 + 10 '*' 2.
Observe how mathematical precedence is obeyed (i.e., the * is worked out before the +).

file <filename>
Prints out the type of data contained in a file. file portrait.jpg will tell you that portrait.jpg is a JPEG image data, JFIF standard. The command file detects an enormous amount of file types, across every platform. file works by checking whether the first few bytes of a file match certain tell-tale byte sequences. The byte sequences are called *magic numbers*. Their complete list is stored in /usr/share/magic. [The word ``magic'' under UNIX normally refers to byte sequences or numbers that have a specific meaning or implication. So-called *magic numbers* are invented for source code, file formats, and file systems.]

free
Prints out available free memory. You will notice two listings: swap space and physical memory. These are contiguous as far as the user is concerned. The swap space is a continuation of your installed memory that exists on disk. It is obviously slow to access but provides the illusion of much more available RAM and avoids the possibility of ever running out of memory (which can be quite fatal).

head [-n <lines>] <filename>
Prints the first <lines> lines of a file or 10 lines if the -n option is not given. (See also tail below).

hostname [<new-name>]
With no options, hostname prints the name of your machine, otherwise it sets the name to <new-name>.

kbdrate -r <chars-per-second> -d <repeat-delay>
Changes the repeat rate of your keys. Most users will like this rate set to kbdrate -r 32 -d 250 which unfortunately is the fastest the PC can go.

lynx <url>
Opens a URL [URL stands for *Uniform Resource Locator*--a web address.]at the console. Try lynx http://lwn.net/.

links <url>
Another text-based web browser.

nohup <command> &
Runs a command in the background, appending any output the command may produce to the file nohup.out in your home directory. nohup
has the useful feature that the command will continue to run even after you have logged out. Uses for nohup will become obvious later.

sleep <seconds>
Pauses for <seconds> seconds. See also usleep.

sort <filename>
Prints a file with lines sorted in alphabetical order. Create a file called telephone with each line containing a short telephone book entry. Then

type sort telephone, or sort telephone | less and see what happens. sort takes many interesting options to sort in reverse (sort -r), to eliminate duplicate entries (sort -u), to ignore leading whitespace (sort -b), and so on. See the sort(1) for details.

strings [-n <len>] <filename>
Writes out a binary file, but strips any unreadable characters. Readable groups of characters are placed on separate lines. If you have a binary
file that you think may contain something interesting but looks completely garbled when viewed normally, use strings to sift out the interesting stuff: try less /bin/cp and then try strings /bin/cp. By default strings does not print sequences smaller than 4. The -n option can alter this limit.

split ...
Splits a file into many separate files. This might have been used when a file was too big to be copied onto a floppy disk and needed to be split into, say, 360-KB pieces. Its sister, csplit, can split files along specified lines of text within the file. The commands are seldom used on their own but are very useful within programs that manipulate text.

tac <filename> [<filename> ...]
Writes the contents of all the files listed to the screen, reversing the order of the lines--that is, printing the last line of the file first. tac is cat backwards and behaves similarly.

tail [-f] [-n <lines>] <filename>
Prints the last <lines> lines of a file or 10 lines if the -n option is not given. The -f option means to watch the file for lines being appended to
the end of it. (See also head above.)

uname
Prints the name of the UNIX *operating system* you are currently using. In this case, LINUX.

uniq <filename>
Prints a file with duplicate lines deleted. The file must first be sorted.

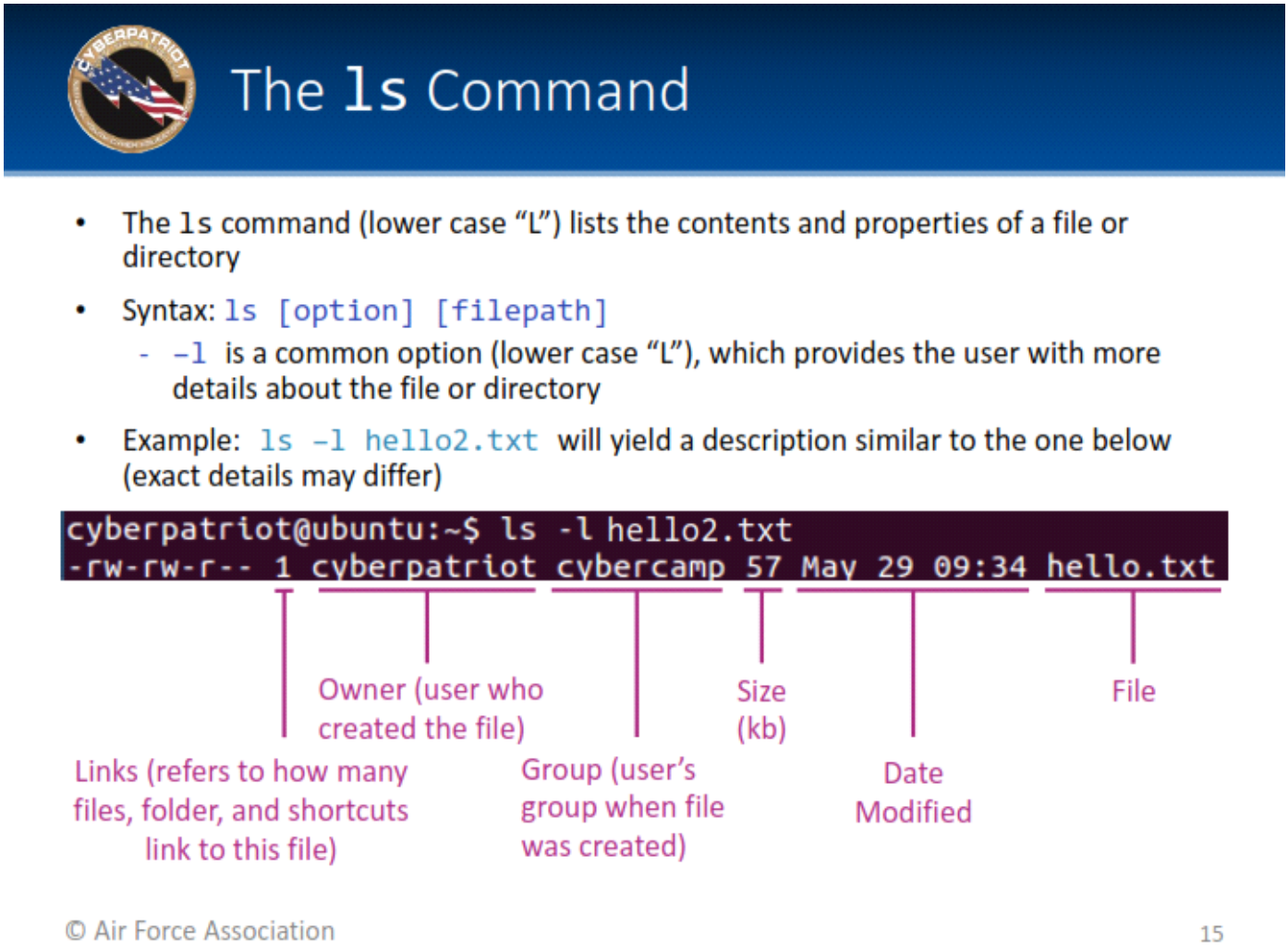
usleep <microseconds>
Pauses for <microseconds> microseconds (1/1,000,000 of a second).

wc [-c] [-w] [-l] <filename>
Counts the number of bytes (with -c for character), or words (with -w), or lines (with -l) in a file.

whatis <command>
Gives the first line of the man page corresponding to <command>, unless no such page exists, in which case it prints nothing appropriate.

whoami
Prints your login name.

Terminating Commands



Viewing File Permissions with the `ls` Command

- File permissions are the first items noted when using the `ls` command with the `-l` option
- File permissions are split into the 10 fields outlined below
- If any fields are blank, the users in that section cannot do that action with the file

1. Type: if this says "d," the item in question is a directory. A blank means it is a file.

2-4. Owner File Permissions: what the user can do with the file or directory

(Blank 2) Read - r

(Blank 3) Write/modify - w

(Blank 4) Execute – x

5-7. Group File Permissions

(Blank 2) Read - r

(Blank 3) Write/modify - w

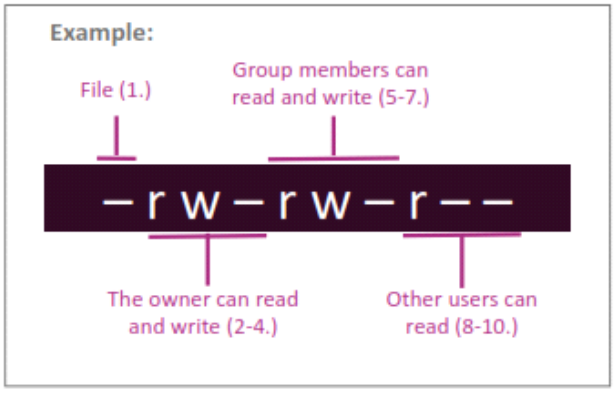
(Blank 4) Execute – x

8-10. Other File Permissions

(Blank 2) Read - r

(Blank 3) Write/modify - w

(Blank 4) Execute – x



© Air Force Association

The chmod Command

- Chmod allows you to change file permissions

Change permissions for the user, group, or others

Add or subtract permissions

Specify whether read, write, or execute privileges are being changed

- Syntax: `chmod [u,g or o][+ or -][r,w, or x] [filepath]`
  - Do not put spaces between the three fields after "chmod"
- Example:
  1. Type `chmod o-r hello2.txt`
  2. Type `ls -l hello2.txt`
  3. If your permissions originally matched those on the last slide, you should see hello2.txt's new file permissions as shown below

```
cyberpatriot@ubuntu:~$ ls -l hello2.txt
-rw-rw---- 1 cyberpatriot cybercamp 57 May 29 09:34 hello.txt
```

Sources: http://condor.depaul.edu/dpowebpg/support/chmod.html, https://help.ubuntu.com/community/FilePermissions

© Air Force Association

Groups

- Work very similarly to Windows
  - Root permissions are required
  1. To list all groups:

     `cat /etc/group`
  2. To add a group:

     `addgroup [groupname]`
  3. To add a user to a group:

     `adduser [username] [groupname]`

```
root@ubuntu: /home/cyberpatriot
root@ubuntu:/home/cyberpatriot# cat /etc/group
root:x:0:
daemon:x:1:
bin:x:2:
sys:x:3:
adm:x:4:syslog,cyberpatriot
tty:x:5:
disk:x:6:
lp:x:7:
mail:x:8:
news:x:9:
uucp:x:10:
man:x:12:
proxy:x:13:
kmem:x:15:
dialout:x:20:
fax:x:21:
voice:x:22:
cdrom:x:24:cyberpatriot
floppy:x:25:
tape:x:26:
sudo:x:27:cyberpatriot
audio:x:29:pulse
dip:x:30:cyberpatriot
www-data:x:33:
backup:x:34:
test:x:1002:cyberpatriot,guest
cybercamp:x:1003:cyberpatriot
root@ubuntu:/home/cyberpatriot#
```

© Air Force Association

Searching from command line

Use **which <name>** to quickly list the path to the file of any command, i.e. firefox or netcat

```
cyber@ubuntu: ~
cyber@ubuntu:~$ which netcat
/bin/netcat
cyber@ubuntu:~$
```

Use **locate <name>** to quickly search all files for that name, i.e. netcat or ssh_config

```
cyber@ubuntu: ~
cyber@ubuntu:~$ locate netcat
/bin/netcat
/etc/alternatives/netcat
/etc/alternatives/netcat.1.gz
/usr/share/doc/netcat-openbsd
/usr/share/doc/netcat-openbsd/README.Debian
/usr/share/doc/netcat-openbsd/changelog.Debian.gz
```

```
cyber@ubuntu:~$ locate ssh_config
/etc/ssh/ssh_config
/usr/share/man/man5/ssh_config.5.gz
cyber@ubuntu:~$
```

Use **whereis <name>** to find the location of a command including all of its associated files and directories.

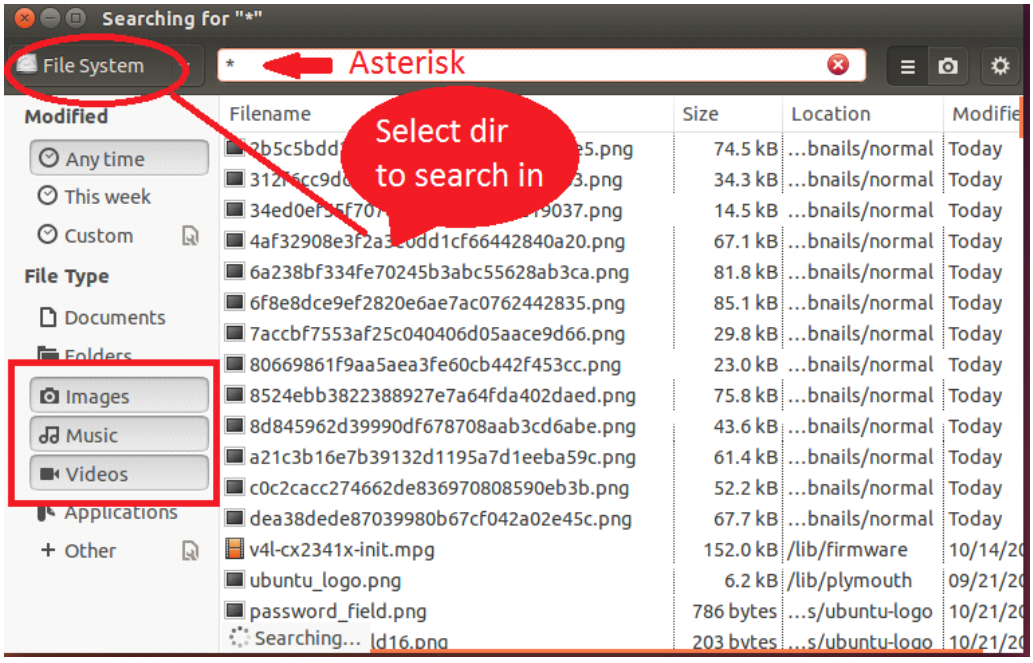
```
cyber@ubuntu: ~
cyber@ubuntu:~$ whereis ssh
ssh: /usr/bin/ssh /etc/ssh /usr/share/man/man1/ssh.1.gz
cyber@ubuntu:~$ whereis netcat
netcat: /bin/netcat /usr/share/man/man1/netcat.1.gz
cyber@ubuntu:~$ whereis firefox
firefox: /usr/bin/firefox /usr/lib/firefox /etc/firefox /usr/share/man/man1/fire
fox.1.gz
```

The *find* command allows for more advanced searches and operations, i.e. search for all files of a specific filetype that was modified within a certain range and then is automatically removed.

To search for multiple media files throughout the entire filesystem, for example, enter:
find / -iname '*.mp3' -or -iname '*.mp4' -or -iname '*.wmv'

```
cyber@ubuntu: ~
cyber@ubuntu:~$ find / -iname '*.mp3' -or -iname '*.mp4' -or -iname '*.wmv'
```

TO SEARCH FOR FILES IN A GUI, install catfish: **sudo apt-get install catfish**


Searching for "*"
File System
*
Asterisk
Select dir
to search in
Modified
Any time
This week
Custom
File Type
Documents
Folders
Images
Music
Videos
Applications
Other
Filename
Size
Location
Modifie
74.5 kB ...bnails/normal Today
34.3 kB ...bnails/normal Today
14.5 kB ...bnails/normal Today
67.1 kB ...bnails/normal Today
6a238bf334fe70245b3abc55628ab3ca.png 81.8 kB ...bnails/normal Today
6f8e8dce9ef2820e6ae7ac0762442835.png 85.1 kB ...bnails/normal Today
7accbf7553af25c040406d05aace9d66.png 29.8 kB ...bnails/normal Today
80669861f9aa5aea3fe60cb442f453cc.png 23.0 kB ...bnails/normal Today
8524ebb3822388927e7a64fda402daed.png 75.8 kB ...bnails/normal Today
8d845962d39990df678708aab3cd6abe.png 43.6 kB ...bnails/normal Today
a21c3b16e7b39132d1195a7d1eeba59c.png 61.4 kB ...bnails/normal Today
c0c2cacc274662de836970808590eb3b.png 52.2 kB ...bnails/normal Today
dea38dede87039980b67cf042a02e45c.png 67.7 kB ...bnails/normal Today
v4l-cx2341x-init.mpg 152.0 kB /lib/firmware
ubuntu_logo.png 6.2 kB /lib/plymouth
password_field.png 786 bytes ...s/ubuntu-logo
Searching...

Profit.

Log Paths

/var/log/messages — Contains global system messages (cron jobs, mail, etc.)

/var/log/syslog — General system logs

/var/log/dmesg — Contains kernel ring buffer information (hardware devices, etc.)

/**var/log/auth.log** — Contains system authorization information (login attempts, etc.)

/var/log/boot.log — Contains information that is logged when the system boots

/var/log/daemon.log — Contains information logged by daemons (services)

/var/log/dpkg.log — Contains information relating to installed/removed packages

/var/log/kern.log — Contains information logged by the kernel

/var/log/lastlog — Displays the most recent login information

/var/log/maillog — Contains log information relating to mail servers

/var/log/user.log — Contains information relating to user level logs

/var/log/cups — Contains information relating to printers

/var/log/cron — Contains information about cron jobs

/var/log/secure — Contains information relating to authentication and authorization (remote logins)

/var/log/wtmp — Contains login records

/var/log/utmp — Contains login records

/var/log/faillog — Contains information relating to failed login attempts

/var/log/httpd or **/var/log/apache2** — Contains web server logs

/var/log/audit — Contains audit logs

/var/log/samba — Contains information relating to Linux-Windows shares

More About the Essentials

UNIX basics - Intro commands, files, and permissions

There are three user categories: owner, group, and others
User: owns the file
Group: users in the group that the file belongs to
World: general public, guests or world

absent write

rwx r-x r-x — In this case... user privs are full; group & world can only read and execute
user group world

Symlinks: symbolic links similar to a shortcut in Windows.

cd <directory>
.. → parent dir
. → current dir
- → last cd

mkdir <directory>

rmdir <directory>

cp <file(s)> <destination>
cp fName.txt ../someDir
cp file1 file2 /home
cp file3 /home/file_txt /var/www
-r → recursively copy
If destination does not exist, it is created
cp *.mp3 ~/Music → cp all mp3 files

mv [opts] <source> <destination>
mv is recursive by default.
-f → do not prompt to overwrite

rm [opts] <filename>
rm * → rm all files in current dir
-r → recursively remove
-i → information prompt to confirm rm

alias <name>="<command> [opts]"
Creates a temporary alias command for the current terminal session only!

touch [opt] <fileName>
touch file2 → creates file2 file
touch file2 → since file2 already exists last modified time will update

- Access to files & folders depends on the user's account.
- All users are presided over the super user "root".
- Each user has absolute control over files that they own.
  - can only be superseded by "root"!
- Files can be assigned to a group of users.
  - allows read, write, execute to be restricted to a subset of users.
- It generally takes root perms to change file ownership.
- *: wildcard means zero or more chars in that spot.

FILES

- A job is the same as a process but is restricted to the processes started by a particular shell.
  - If shell closes, jobs will terminate.

- A job is the same as a process but is restricted to the processes started by a particular shell.
  - If shell closes, jobs will terminate.
  - Jobs can only be foreground or background.
  - Only one foreground job at a time!

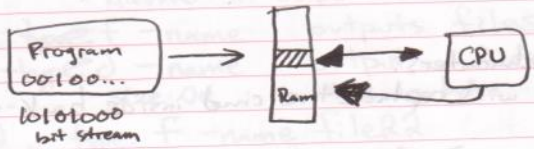
jobs -l	lists active jobs and its PID

- Add & to the end of a command to run job in background.

Kill -9 %id	Kills by job id. Replace ID with a number.
fg %id	Resumes job "id" in the foreground.
Ctrl + Z	Suspend a process.
bg %id	Resumes job "id" in the background.

- A process is a program in execution.
  - Edit files with nano
  - Listening to music with vlc

- A process is a program in execution.
  - Edit files with nano
  - Listening to music with vlc
- A process consists of resources that the program needs to run.
  - Such as instructions, file descriptors, memory to be used, IPC tools, etc.
- The OS keeps track of processes by assigning each a process ID (PID).



- An important process that is always present since startup: init → PID of 1

ps	Process status. will list PIDs for processes owned by shell.
ps -aux -fae	Shows full information about all processes on BSD machines.
ps -U user	Processes belonging to a user.
Kill	sends diff level signals to a process to request shutdown and exit.
Kill -9 PID	force kill a PID
Kill -19 PID	suspends a PID
Kill -list	list all signals

find dirNameToSearch -name targetFile -print
└ Find will search dirNameToSearch recursively for targetFile.

find /home -name "*.txt" -print 2>/dev/null
- /dev/null is a special file which behaves like a blackhole.
  It consumes what we send to it as trash. So 2>/dev/null will redirect error messages to /dev/null.

find dirNameToSearch -name targetFile -print

└ Find will search dirNameToSearch recursively for targetFile.

find /home -name "*.txt" -print 2>/dev/null

- /dev/null is a special file which behaves like a blackhole.
  It consumes what we send to it as trash. So 2>/dev/null will redirect ☆ error messages to /dev/null.

-name outputs both files and directories
-type f -name outputs files
-type d -name outputs directories
-size 0c size in bytes. c denotes characters, replace it with K for kilo bytes.

find . -type f -name file22

└ searches current dir for file named file22. Will output only files.

find . -name -size +100k file22

Searches current dir for file22 that is greater than 100k.

find . -name -size -100K *

Searches current dir for all files less than 100k in size.

find . -name "*.txt" -exec wc -l {} ';' ☆

Searches for all .txt files and counts the amount of lines.

stdin if no file

grep [options] PATTERN [FILE]	- locates particular content within files.
grep CD catalogue	- Searches for PATTERN in all files FILE
grep 'Compact' catalog	If one line in catalog ends with word 'Compact' [illegible] with 'disc', grep will not

stdin if no file

grep [options] PATTERN [FILE]	- locates particular content within files.
grep CD catalogue	- Searches for PATTERN in all files FILE
grep 'Compact' catalog	
grep -i 'Compact' catalog	If one line in catalog ends with word 'Compact' and the next begins with 'disc', grep will not match either line. -i means case insensitive
grep -n	Prints lines found within files & line numbers.
grep -r	Search a given dir recursively.
* grep -rn dbConnect --include=*.c ./	Searches all c files recursively for func called dbConnect and will print out the line numbers where found. searches current dir
grep -v	Inverts sense of matching to select non-matching lines.
grep -c	Displays total number of matching lines.
grep -x	Pattern must match a line directly.

ls -LR | grep errfile
- outputs all files and dirs recursively
- then we look for a file whose name containing 'errfile' in the output.
- -L follows links

who | grep wade
- see if wade is logged in

tail -n8 a_file | grep "boo"
- -n8 and -8 the same
- Searches last 8 lines of a_file for "boo"

Regular Expressions

A regular expression is a text string that contains metacharacters to specify a group of patterns to be matched. Most characters represent themselves.

Matches any one character with the exception

Regular Expressions

A regular expression is a text string that contains metacharacters to specify a group of patterns to be matched. Most characters represent themselves.

.	.wn	Matches any one character with the exception of newline. This will match "wn" preceded by a character.
*	-*	Matches the previous character 0 or more times. This will match "-", "--", "---", etc.
$	a$	Matches the end of the line. This matches lines that end with letter a.
^	^a	Matches the beginning of the line. This matches lines whose first letter begins with a.
[ ]	[abc]	Encloses a character set and matches any member of the set. This matches a or b or c. A hyphen - can specify ranges.
[^]		Match characters <u>not</u> listed in the class.
\	\.$	Before a metacharacter when you want to specify that literal character. In this case, the dot in this pattern is treated literal and is not special, but $ remains special. So \.$ matches lines that end with a dot.

Th and F, meet 208 @ HPM

Metacharacters

Shell shortcuts

- Tab – automatically completes pathnames and commands (autocompletion)
- ↑ and ↓ – browse through command history
- Ctrl + e – jump cursor to the end of line
- ctrl + a – jump cursor to the beginning of line
- ★ ! <command> – repeat a command w/ bang operator ~~(most recent command)~~ most recent like <command>
- ★ !! ~~<command>~~ – repeat the most recent command that was entered

Shell Expansion / Metacharacters

*	Matches any string, including null (i.e. 0 or more characters)
^	matches any characters NOT inside square brackets. characters inside brackets provides options for only one place.
?	Matches a single character
.	
{ }	
[ ]	matches any one character, use a dash for ranges, can have commas between characters and ranges

Example	Matched	Not matched
L*ure*	Lecture2.pdf Lectures/	Lectuuree .Lecture
ca?	cat can cap	ca cake
[SL]ec*	Lecture Section	Vector.tex
Day[1-4].pdf	Day1.pdf Day2.pdf	Day5.pdf Day.pdf
[A-Z,a-z][0-9].mp3	A9.mp3 z4.mp3	B29.mp3 9a.mp3
[^A-P]ec*	Section.pdf	Lecture.pdf
[^A-Z a-z]*	9Days.avi .bashrc	vacation.png
[^A-Z,a-z]*	9Days.avi .bash_profile	vacation.png

★ If we have file1 file2 file3 file22 file33, to delete all except file22 and file33: rm *[a-z][0-9]

Shell commands can be:

1. Executable programs
2. Shell scripts (must have +x permission)
3. Builtin shell commands

Env Variables

PATH
HOME
JAVA_HOME
SHELL
LD_LIBRARY_P

Permissions and Files

* chmod <mode> <file>	Change perms of file based on mode
ugotrwx	<mode> is a combination of 3 fields.
	u = user, g = group, o = others, add or rm, perms
a-r	Removes read access from everybody

Mode can be symbolic or numeric

Sym mode		
ug	t or -	rwx
∧	∧	∧
types of users	add or rm	perms

Sym: a, u, g, o
Num: 3 digits of octals, e.g. 755

Num mode	
7 5 5	Read = 4
∧ ∧ ∧	Write = 2
user group world	Execute = 1

Permissions	user	group	world
rwx r-x r-x	rwx	r-x	r-x
-wx --x ---	-wx	--x	---
r--w---x	r--	-w-	--x

Example
User perms: r $2^2=4$ w $2^1=2$ x $2^0=1$ = 777
Group perms: r 0 w 0 x 1 = 310
World perms: r 0 w 0 x 1 = 420

Sym mode changes perms incrementally, while num mode specifies final perms.

Type of file	Description
Text	Human readable and editable
Binary	Commonly used as executables, libs, media, archives, and not human readable

- Shell can interact with text files.
  - nano <filename>
- Print information to terminal
  - cat <filename1> <filename2>
  - cat <file>

Scroll through 1 pg at a time
- more <filename>

Read x lines at beginning
- * head -[numlines] <filename>

Read x lines at end
- tail -[numlines] <filename>

→ Default numlines = 10

less <filename>
- G: end of file
- g: beginning of file
- j: scroll down
- k: scroll up
- q: quit
- h: help
- /pattern: search for pattern in file

UNIX basics – intro commands, files, and permissions

ls → list files and dirs
pwd → print working dir
cd → change dir
clear → clear the screen

ps -p $$ → echo the shell
groups → see what group I'm in

ls -l → long list files and dirs

- Blue = dir
- Black = ordinary file
- Green = binary or executable file

* Relative: relative to the dir you are in, e.g. if in /home/user, and there is a "Doc" dir in /home/user, then: cd Doc

* Absolute: full path starting from root /

Show help information
man <CommandName>
info <CommandName>
help <CommandName>

RTFM!

Shows manual for computer progs (system files & configs)

ls [options] [file] lists directory contents.

- -l long list (detailed)
- -a show all files (including hidden dotfiles)
- -h human readable file size

Type: d for dir, - for file, l for symlink; in bytes; last modified

d rwx-xr-x	3	will	finance	1034	Nov 20 10:45	will
perms	links	owner	group	size	date	name

Permissions are a set of characters describing access rights.
There are 3 access types for 3 groups for a total of 9 chars.

- "R" → read
- "W" → write
- "X" → executable

More chmod, chown, and chgrp Goodness

Linux is designed to be a multi-user system and as such it is important for each user and group's information to be protected from all of the other users and groups.

Therefore, dangerous file permissions or malicious file owners can be a problem. For instance, it would be a very bad idea to let all users, group, and world have complete read, write, and execute access to important configuration files like /etc/ssh/sshd_config

The following is a basic summary of what you need to know:

1. Every file on Linux has an owner.
2. Every file on Linux has permissions associated with various owners (user owner, group owner, world owner).
3. You can change the user owner of a file with the **chown** (change owner) command.
4. You can change the group owner of a file with the **chgrp** (change group) command.
5. You can change permissions (the mode) associated with various owners with the **chmod** (change mode) command.

If you understand the five above statements, then all you need to know is how to use commands 3-5 and you will be able to better manage you filesystem.

To better understand the above 5 statements and utilize the commands, let us setup a hypothetical situation:

Suppose we are logged in as the user "Adam" and run the command **touch file.txt** (which will create a new file called "file.txt")

If we run **ls -l**, we can see that **file.txt** belongs to the user "Adam," because it was the user "Adam" who created the file.

Now, let us change the permissions of this file with Step #5, or, the **chmod** command.

chmod 770

What does this mean?

The first "7" means that the **USER OWNER** has the following permissions: Read, Write, Execute
The second "7" means that the **GROUP OWNER** has the following permissions: Read, Write, Execute
The third "0" means that the **WORLD OWNER** has the following permissions: None

The USER OWNER is the user account that owns the file.
The GROUP OWNER is all the users belonging to the group that is associated with the file.
The WORLD OWNER is all the users who are not the USER OWNER and are not the GROUP OWNER.

Based on this information, we can see that the **USER** and **GROUP** owners have complete access over the file.

Since "Adam" is the **USER OWNER** of "file1.txt", "Adam" has full permissions over the file.

Let us assume that "file1.txt" belongs to the **GROUP OWNER** called "staff".

Supposed we login as a new user named "Tom" who is part of the "user" group.

Tom navigates to */home/Adam*
and notices a new file called "file1.txt"

Curious, Tom attempts to read the file. Unfortunately for Tom, Tom is unable to open the file and receives a "Permission Denied" error.

WHY? Because Tom is part of the "user" group. Since "file1.txt" belongs to the "staff" group, and since Tom is not the **USER OWNER** of that file (which is Adam), then Tom is seen by the file as being part of the **WORLD OWNER**.

If we recall, the **WORLD OWNER** received a 0, which means no permissions. This is why Tom received a permission denied error!

Now let is supposed we login as a new user named "Samantha" who is part of the "staff" group.

Like Tom, Samantha traverses to */home/Adam*
and notices the file called "file1.txt"

Samantha attempts to open the file and is successful, unlike Tom.

WHY? Because Samantha is part of the "staff" group. Since "file1.txt" belongs to the "staff" group, Samantha is seen as having the permissions that are associated with this group, which happens to be 7 (read, write, execute).

Let us assume that Samantha tries out the **chgrp** command and changes the **GROUP OWNER** of Adam's file to "user"

chgrp user file1.txt

After making this change, Samantha realizes her mistake and tries to open up the file again. Unfortunately for her, she is no longer able to because the **GROUP OWNER** is set to "user," which Samantha is not. Samantha is also not the **USER OWNER**, that is Adam. Therefore, the file now recognizes Samantha as a **WORLD OWNER** who has no permissions.

Now let us return to Tom's console. Tom is now able to read, write, and execute "file1.txt" even though Tom was not able to do so in the past. This is because Samantha changed the **GROUP OWNER** of the file to the same group that Tom belongs to.

Now let us return to Adam's console. Even though the **GROUP OWNER** of the file has changed, Adam still has full access to "file1.txt" because the **USER OWNER** was unchanged and the **USER OWNER** is Adam. In contrast, if we assume that Samantha was being super devious and instead decided to run the command

chmod 470 file1.txt && chgrp user file1.txt

Then Adam would only be able to read the file and could not make any modifications to the file, delete the file, or even execute the file. Poor Adam!

This leads us to the conclusion: how do I know what numbers to use when changing the ownership permissions of a file?

Remember that the first number = **USER OWNER**
The second number = **GROUP OWNER**
The third number = **WORLD OWNER (public)**

0 = Nothing
1 = Execute
2 = Write
4 = Read

You can add the above numbers together to combine the permissions. Some examples:

1+2 = 3 = Execute and Write only
2+4 = 6 = Read and Write only
1+2+4 = 7 = Read and Write and Execute

So when we combine all of this information, we can see the examples:

chmod 764 file1.txt
USER OWNER = 7 = Read and Write and Execute
GROUP OWNER = 6 = Read and Write only
WORLD OWNER = 4 = Read only

chmod 700 file1.txt
USER OWNER = 7 = Read and Write and Execute

GROUP OWNER = no permission
WORLD OWNER = no permission

chmod 644 file1.txt
USER OWNER = 6 = Read and Write only
GROUP OWNER = 4 = Read only
WORLD OWNER = 4 = Read only

Chapter 1.3
INCIDENT RESPONSE PREPARATION

Incident Response Toolkit

FIREWALL

ufw	*sudo apt-get install ufw*
csf	*wget http://configserver.org/free/csf.tgz*
iptables	*man iptables*

MALWARE DETECTION

lmd	*wget http://rfxn.com/downloads/maldetect-current.tar.gz*
bitdefender	*sudo apt-get install bitdefender-scanner-gui*
chkrootkit	*sudo apt-get install chkrootkit*
rkhunter	*sudo apt-get install rkhunter*

SEARCH

Tool	Description	Example
locate	The quickest method of finding the locations of files and dirs	*locate "*.png"*
which	Quickly search the entire system to find where a command lives	*which netcat*
whereis	Similar to *which* but displays a more verbose output	*whereis firefox*
find	The Swiss Army Knife of Linux search. Understanding this cmd has many advantages	*find . -name "*.txt" -exec cat {} \; | wc -l*
catfish	A graphical user interface that utilizes the *locate* and *find* commands	*catfish*

INTRUSION DETECTION

tripwire	*sudo apt-get install tripwire*
snort	*sudo apt-get install snort*

SCANNING && HARDENING

bastille	*sudo apt-get install bastille*
selinux	*sudo apt-get install selinux*
libpam-cracklib	*sudo apt-get install libpam-cracklib*
fsaudit	*wget http://www.cise.ufl.edu/~jfh/jst/fsaudit*
COPS	*wget ftp://ftp.cerias.purdue.edu/pub/tools/unix/scanners/cops/cops.1.04.tar.gz*
Lynis	*sudo apt-get install lynis*
tiger	*sudo apt-get install tiger*
OpenVAS	*sudo add-apt-repository ppa:openvas/openvas6 && sudo apt-get update && sudo apt-get install openvas-manager*

	openvas-scanner openvas-administrator openvas-cli greenbone-security-assistant sqlite3 xsltproc texlive-latex-base texlive-latex-extra texlive-latex-recommended htmldoc alien rpm nsis fakeroot

Service List

Below is a list of common services that can be found on linux. Each are ranked from 1-5 regarding how malicious the tool would be during incident response. Note, just because a tool is ranked 1 does not mean that it is safe or acceptable to maintain--see your scenario!

apache2 [1] -- http web server
bind9 [1] -- domain name system server
inspircd [4] -- internet relay chat server
maltego [1] -- reporting tool
mysql-server [1] php5-mysql [1] -- sql database server
nginx [1] -- http web server
openssh [3] -- secure remote communication protocol
php5 [3] -- server-side scripting language used with a web server
proxychains [3] -- tunneling tool
samba [2] -- reimplementation of SMB protocol to share access to resources
scalpel [3] -- forensic carving tool
telnet [4] -- insecure remote communication protocol

Hacking Tools

acccheck
ace-voip
Aircrack-ng
Amap
android-sdk
apache-users
apktool
Arachni
Arduino
Armitage
Asleap
Automater
Backdoor Factory
BBQSQL
BED
BeEF
bing-ip2hosts
Binwalk
BlindElephant
Bluelog
BlueMaho
Bluepot
BlueRanger
Bluesnarfer
braa
bulk-extractor
Bully
Burp Suite
Capstone
CaseFile
CDPSnarf
CeWL

IDEA: What if we could automate the process of traversing through this list to search for all of the hacking tools in one command?

Included here a .txt of the list of Hacking Tools (unique entries sorted A…Z) hacktools

Method: One of the faster approaches would be to quickly code a bash script.
Objective: Read the contents of a file, and for each line, run a search command (like find) against that line.
Solution: http://stackoverflow.com/a/1521498

TIP: Click on a hacking tool in the table on the left to learn more information about that tool. An Internet connection is required.

chntpw
cisco-auditing-tool
cisco-global-exploiter
cisco-ocs
cisco-torch
CmosPwd
Cookie Cadger
copy-router-config
coWPAtty
crackle
creddump
crunch
CryptCat
Cuckoo
CutyCapt
Cymothoa
DAVTest
dbd
DBPwAudit
dc3dd
ddrescue
deblaze
dex2jar
DFF
DHCPig
DIRB
DirBuster
diStorm3
DMitry
dnmap
dns2tcp
DNSChef
dnsenum
dnsmap

DNSRecon
dnstracer
dnswalk
Doona
dos2unix
DotDotPwn
Dradis
Dumpzilla
eapmd5pass
edb-debugger
enum4linux
enumIAX
exploitdb
extundelete
Fern Wifi Cracker
Fierce
fiked
fimap
findmyhash
Firewalk
Foremost
fragroute
fragrouter
FunkLoad
Galleta
Ghost Phisher
GISKismet
GoLismero
goofile
gpp-decrypt
Gqrx
Grabber
Greenbone Security Assistant
gr-scan

GSD
Guymager
hamster-sidejack
hash-identifier
HexInject
HexorBase
hping3
HTTPTunnel
http-tunnel
iaxflood
Inguma
Intersect
InTrace
Inundator
inviteflood
iPhone Backup Analyzer
ipv6-toolkit
iSMTP
isr-evilgrade
jad
javasnoop
jboss-autopwn
JD-GUI
John the Ripper
Johnny
joomscan
jSQL
kalibrate-rtl
KeepNote
keimpx
KillerBee
Kismet
lbd
Linux Exploit Suggester

Lynis
MagicTree
Maltego Teeth
Maskprocessor
masscan
mdk3
Metagoofil
mfcuk
mfoc
mfterm
Miranda
mitmproxy
multiforcer
Multimon-NG
Ncrack
Nipper-ng
Nishang
Nmap
ntop
oclgausscrack
ohrwurm
OllyDbg
openvas-administrator
openvas-cli
openvas-manager
openvas-scanner
Oscanner
p0f
PACK
PadBuster
Paros
Parsero
patator
pdfid

pdf-parser
pdgmail
peepdf
phrasendrescher
pipal
PixieWPS
plecost
polenum
Powerfuzzer
PowerSploit
protos-sip
ProxyStrike
pwnat
RainbowCrack
rcracki-mt
Reaver
rebind
Recon-ng
redfang
RegRipper
responder
RidEnum
RSMangler
RTLSDR Scanner
rtpbreak
rtpflood
rtpinsertsound
rtpmixsound
Sakis3G
sbd
sctpscan
SET
sfuzz
ShellNoob

SidGuesser
SIPArmyKnife
SIPp
SIPVicious
Skipfish
SlowHTTPTest
smali
smtp-user-enum
SniffJoke
snmpcheck
Spooftooph
SQLdict
sqlmap
Sqlninja
sqlsus
sslcaudit
SSLsplit
sslstrip
SSLyze
Statsprocessor
t50
Termineter
THC-Hydra
THC-IPV6
THC-pptp-bruter
THC-SSL-DOS
theHarvester
TLSSLed
tnscmd10g
TrueCrack
twofi
U3-Pwn
ua-tester
Uniscan

unix-privesc-check
URLCrazy
Valgrind
Vega
VoIPHopper
Volatility
w3af
WebScarab
Webshag
Webshells
WebSlayer
WebSploit
Weevely
Wfuzz
Wifi Honey
Wifitap
Wifite
Winexe
Wireshark
WOL-E
wordlists
WPScan
Xplico
xspy
XSSer
YARA
Yersinia
zaproxy

General Checklist

- [ ] Forensics questions, verify answers, and gather paper+pen[cil].
- [ ] First thing's first -- change passwords, manage accounts, update, and enable a firewall.
- [ ] Security policies and insecure defaults.
- [ ] Patches and hardening.
- [ ] Search and Destroy.
- [ ] Intrusion detection and network monitoring + other.

Specific Checklist

- [ ] Do the forensics questions.
- [ ] Verify that the forensics questions are answered correctly.
- [ ] For all points you get, write down your answers and the changes you made on paper!

- [ ] Change passwords of **ALL** authorized user accounts, unless stated otherwise by the scenario.
- [ ] Switch *unauthorized users from administrative groups to standard.
- [ ] Delete any malicious users that are not authorized to be on the system [re-read scenario].
- [ ] Install updates and upgrades.
- [ ] Set automatic updates to daily.
- [ ] Check the *sudoers* file for unauthorized users **AND** groups. ****use visudo**
- [ ] Enable a firewall.
- [ ] Disable the guest account.
- [ ] Ensure console security.
- [ ] Limit access to *su*.
- [ ] Disable IPv6 if not needed.
- [ ] Reconfigure shared memory.
- [ ] Restrict DMESG.
- [ ] Secure your home directory 0700.
- [ ] Lock all accounts with no password.
- [ ] Make sure that no non-root accounts have UID set to 0.
- [ ] Harden Linux Kernel parameters.
- [ ] Disable unwanted SUID and SGID bins.
- [ ] Install and enable cracklib.
- [ ] Set a minimum password length of 8.
- [ ] Enforce account lockout policy.
- [ ] Enable password history.
- [ ] Enable password expiration.
- [ ] Enable auditing.
- [ ] Automate hardening tasks.
- [ ] Harden *SSH* config.
- [ ] Remove unneeded services [re-read your scenario very carefully].
- [ ] Patch heartbleed.
- [ ] Patch shellshock.
- [ ] Enable apparmor.
- [ ] **IF** *apache2* (web server) is needed, **THEN** harden apache2 config.
- [ ] **IF** a web server AND php is needed, **THEN** harden the php config.
- [ ] Remove dangerous files from /var/www.
- [ ] **IF** *ssl* is needed with apache2 (big IFF, read scenario), **THEN** ensure a secure SSL config. Mitigate BEAST, disable SSLv3, and enable HSTS.
- [ ] **IF** *bind** (DNS server) is needed, **THEN** harden bind*. Otherwise, remove.
- [ ] **IF** *mysql* (sql server) is needed, **THEN** harden mysql. Otherwise, remove.
- [ ] **IF** *dovecot* (email server) is needed, **THEN** maintain. Otherwise, remove.
- [ ] **UNLESS** needed as stated in your scenario, remove FTP, telnet, rlogin and rsh services.
- [ ] Ensure that **ALL** required services are not being blocked by firewall. **DENY** all other traffic in firewall.
- [ ] Clean DNS resolver.
- [ ] Secure home directories.
- [ ] Clear the issue banner.
- [ ] Search for writable files by other and set more strict permissions as necessary.
- [ ] Check cronjobs.
- [ ] Check startup directories for malicious files.
- [ ] Find files with unwanted permissions.
- [ ] Remove unnecessary media files.
- [ ] Search for *netcat* and *nc* and remove them.
- [ ] Secure ports.
- [ ] Install an AV and run a virus scan.
- [ ] Install an anti-rootkit tool and and run a rootkit scan.
- [ ] Install *tiger* and run a scan.
- [ ] View tiger log, use *tigexp* to investigate, and correct potential warnings.
- [ ] Check and monitor logs with either *logwatch* or *logcheck*.
- [ ] Install *snort* and *snorby* for an intrusion detection system.

- [ ] Install Ubuntu Server Secure script.
- [ ] Install *psad* [port scan attack detector].
- [ ] **IF AND ONLY IF** your scenario says so, run a distribution upgrade.
- [ ] **IF** all else fails **AND** you are in *dire need of points, consider recompiling the linux kernel with *grsec*.

Extras Checklist

- ☐ Re-install core utilities: sudo apt-get install coreutils
- ☐ Update the kernel [read the scenario first!]: sudo apt-get dist-upgrade
- ☐ Clean DNS resolver: update the file /etc/resolv.conf with 8.8.8.8
- ☐ Install task manager GUI: sudo apt-get install gnome-system-monitor
- ☐ Clear the banner text inside of /etc/issue.net
- ☐ Use nmap to scan for open ports: nmap -n -PN -sT -sU -p- 127.0.0.1
- ☐ Secure home directories: sudo chmod 750 /home/username

Example Solutions

These are some example solutions from the past.

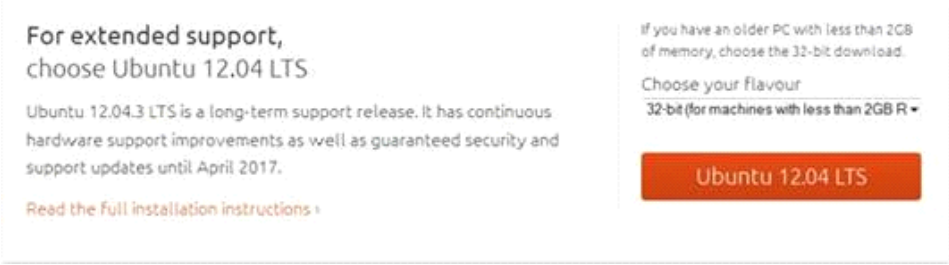
- Linux kernel has been updated -- sudo apt-get dist-upgrade
- Removed unauthorized user
- User lumpyspaceprincess is not an administrator
- Changed insecure root passwd -- sudo passwd -l root
- Guest account disabled
- Install updates from important security updates
- Sudo has been updated
- Prohibited MP3 files removed
- DNS service (bind9) is disabled or removed
- A firewall has been enabled
- SSH root login has been disabled (/etc/ssh/sshd_config)
- mysql remote access has been disabled
- System checks for updates daily
- PHP backdoor found (/var/www/cmd.php)
- Removed nc backdoor
- Bad permissions fixed for phpBB (/var/www/config.php on line 660)
- Removed samba
- Removed smb
- openbsd-inetd server has been disabled
- Disabled automatic login (sudo sed -i.backup -e 's/autologin-user=\(.* \)/autologin-user=/ /etc/lightdm/lightdm.conf)
- NFS server has been disabled
- Insecure sudoers configuration fixed

Chapter 1.4
SETTING UP THE VM

Setting up the VM

Virtualization allows a computer user to run multiple operating systems and applications on one host system. Imagine a scenario in which you wanted to run Windows and Mac OSX at the same time. Most people would think that this task requires two computers: one computer that has windows on it, and another that has OSX on it. Thanks to virtualization, we can avoid needing to have that second computer. Simply put, we could run Windows XP, Windows 7, Mac OSX, Ubuntu Linux, and Fedora Linux all at the same time on one machine.

1. Download Ubuntu 12.04LTS. This can be located at: http://www.ubuntu.com/download/desktop

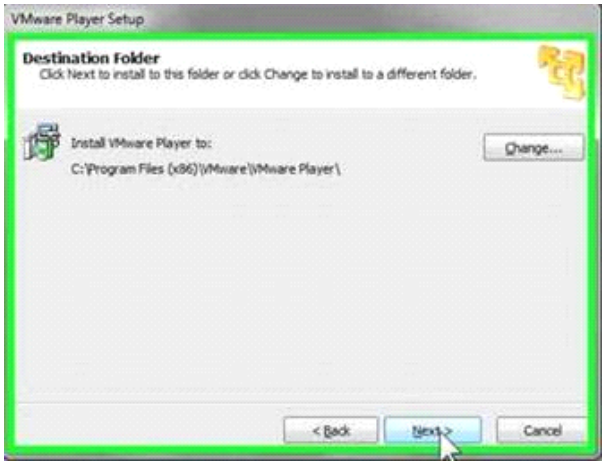


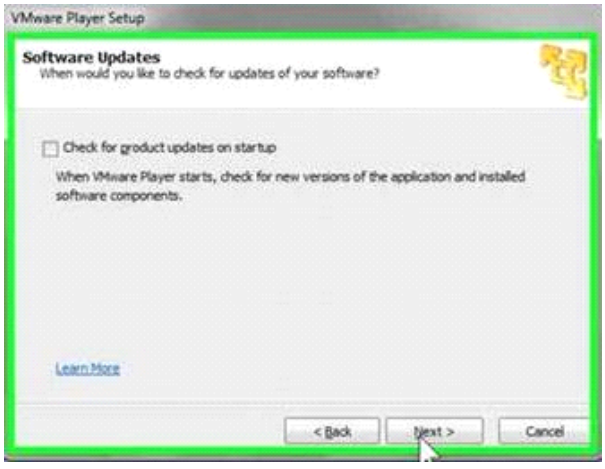
Select 32-bit.

2. Download VMware Player. An alternative to *VMware Player* is *Virtualbox*, which some may enjoy for *Seamless Mode*. The competitions generally use *VMware*, so it might be wise to get familiar with using this virtualization software. This can be located at:
https://my.vmware.com/web/vmware/free#desktop_end_user_computing/vmware_player/6_0

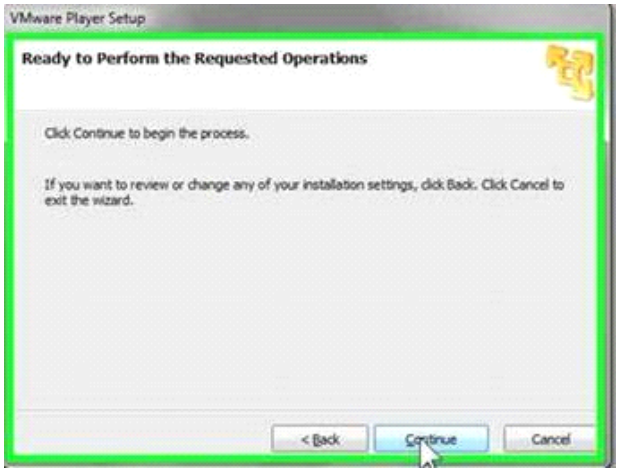
3. Install VMware Player.

VMware Player Setup
License Agreement
Please read the following license agreement carefully.
VMWARE END USER LICENSE AGREEMENT
PLEASE NOTE THAT THE TERMS OF THIS END USER LICENSE AGREEMENT SHALL GOVERN YOUR USE OF THE SOFTWARE, REGARDLESS OF ANY TERMS THAT MAY APPEAR DURING THE INSTALLATION OF THE SOFTWARE.
IMPORTANT-READ CAREFULLY: BY DOWNLOADING, INSTALLING, OR USING THE SOFTWARE, YOU (THE INDIVIDUAL OR LEGAL ENTITY) AGREE
I accept the terms in the license agreement.
I do not accept the terms in the license agreement.
Print
< Back
Next >
Cancel


VMware Player Setup
Destination Folder
Click Next to install to this folder or click Change to install to a different folder.
Install VMware Player to:
C:\Program Files (x86)\VMware\VMware Player\
Change...
< Back
Next >
Cancel

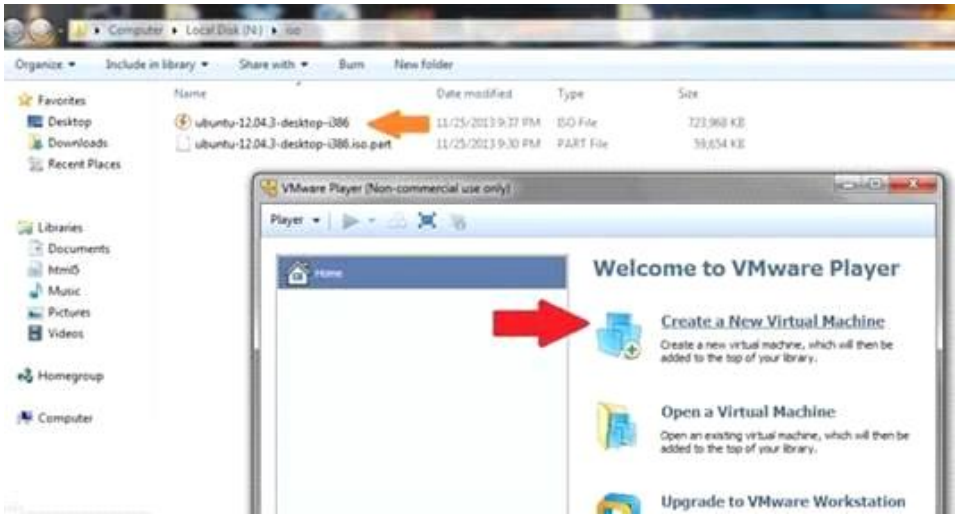

VMware Player Setup
Software Updates
When would you like to check for updates of your software?
Check for product updates on startup
When VMware Player starts, check for new versions of the application and installed software components.
Learn More
< Back
Next >
Cancel

VMware Player Setup
User Experience Improvement Program
Would you like to send feedback to VMware?
Help improve VMware Player
Send anonymous system data and usage statistics to VMware.
Learn More
< Back
Next >
Cancel

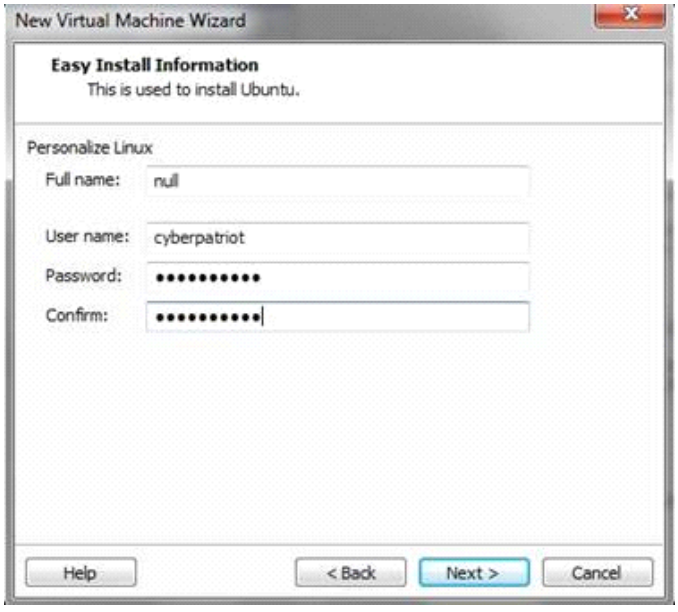


4. Create a new VM with the Ubuntu 12.04LTS iso.

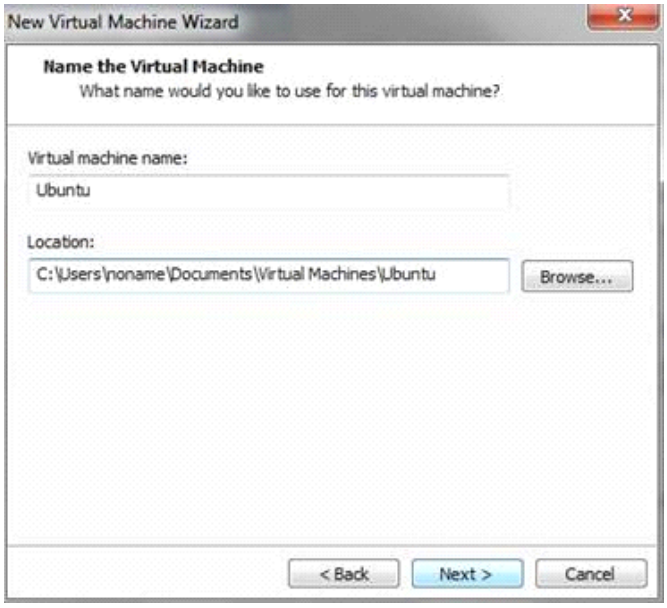
Open up VMware Player from the menu and select “Create a New Virtual Machine.”



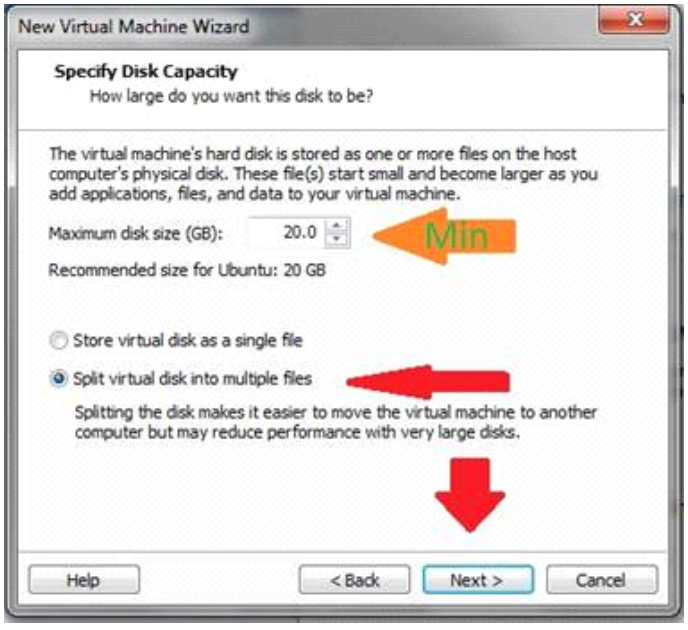
Select the .iso you just downloaded and hit “Next”. To do this, select the “Installer disc image” radio button, hit “Browse,” and then find “Ubuntu-12.04.3-desktop.iso.”



Full Name: Not required. Avoid using full name for confidentiality
User name: This will be your login name, so make sure it is memorable
Password: This is the password that goes with your username

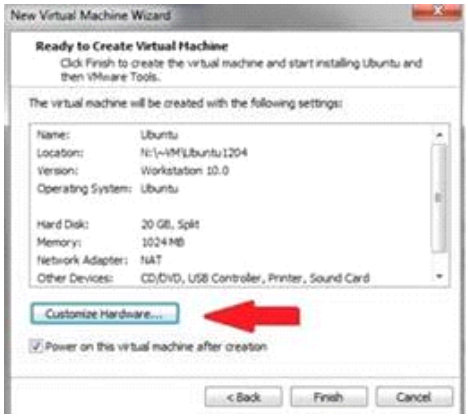


The *virtual machine name* is the name of the VM, which will be displayed in an option list upon starting VMware Player. The *location* is the absolute path to where the files of your virtual machine will be stored. Depending on your configuration, these images might use up anywhere between (20GB,X) amount of space, so plan accordingly. If you don't intend to save the virtual machine to an external hard drive and are comfortable with the default location, simply name the *virtual machine name* to your liking and hit "Next >."

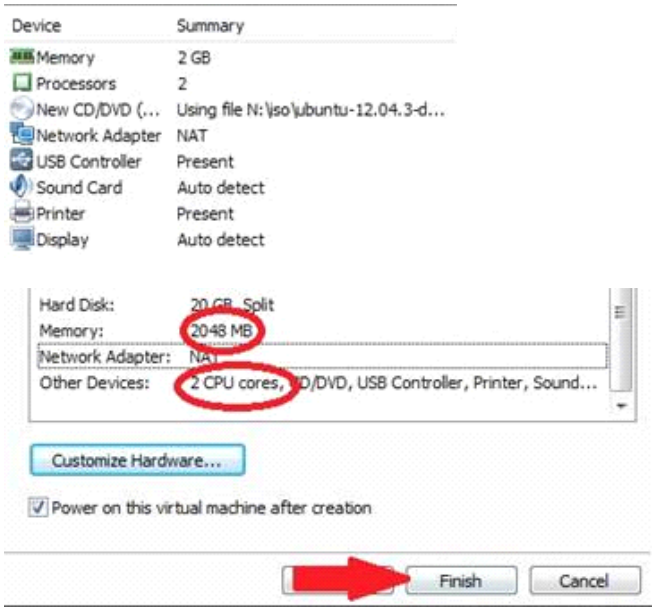


Give yourself 20GB **minimum** disk space and split the virtual disk into multiple files. Select "Next >" to continue.

Before hitting "finish," select "customize hardware."



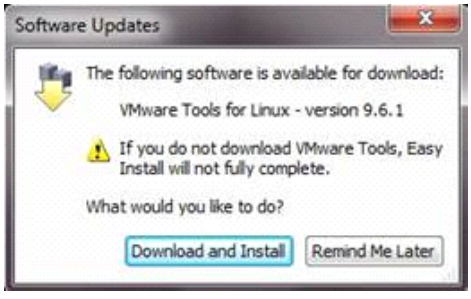
Give yourself 2GB of memory and 2 cores minimum. During competition, you should probably use 3GB of memory.



Now that we can see the changes are reflected, hit “Finish!” The virtual machine should automatically power on after doing so.

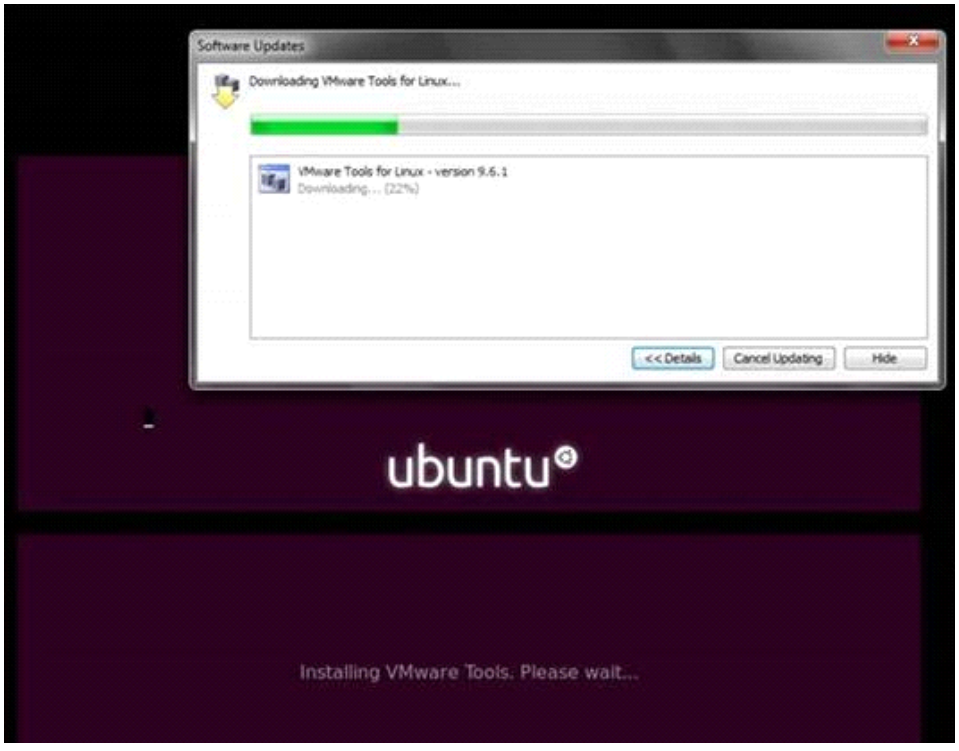
5. Create a new VM with the Ubuntu 12.04LTS iso.

IF you are prompted to install “VMware Tools for Linux,” do so. Select “Download and Install.”



Meanwhile, Ubuntu will install itself within the virtual environment. The amount of time this takes will vary.

Notice: When you click inside the VM, your cursor will lock inside that environment. If you need to move the cursor out of the virtual machine, press Ctrl+Alt



Downloading and installing *VMware Tools for Linux* may take a while. It may or may not be necessary to install depending on user preferences. Why VMware Tools? Here are a few reasons:

- Installs additional device drivers.
- Installs a service that allows for cleaner shutdown, restart, etc.
- Creates "vmware-user" service, which allows for the user to drag and drop files between the host machine and virtual machine.

After Ubuntu has installed, you should be prompted to login.

IF you are unable to get to this interface, hit **Ctrl+Alt+F1**, login, and type in the command **startx**

To enter commands in the graphical user interface, open up the

menu and search for “terminal.”

For the remainder of this guide, we will not use the GUI. It is best to get accustomed to being able to enter in commands without having the graphical user interface available, as there will be situations where there won’t be.

Notice that when you hit **Ctrl+Alt+F1**, you leave the GUI and enter into a different terminal session. You can traverse through the terminal lines by pressing Ctrl+Alt+F_x. Your GUI is generally available on tty7 (Ctrl+Alt+F7).

Take advantage of multiple terminal sessions. You can be running updates in tty1, scanning your system with TIGER in tty2, and installing CSF in tty3 all at the same time.

Chapter 2.
FIRST THING'S FIRST

passwd

Change the password.

To do this, use the **passwd** command.

```
w@localhost:~$ passwd
Changing password for w.
(current) UNIX password:
Enter new UNIX password:
Retype new UNIX password:
```

You can also change the password of any user account by stating the username after the command. Example: **passwd bob**

Be sure to also change the password for the **root** account and use secure passwords! You can use apg to generate secure passwords on your machine. Use a minimum password length of 12!

sudo apt-get install apg

apg –m 12 –s –t –M SNCL

```
w>@ubuntu:~$ apg -m 12 -s -t -M SNCL

Please enter some random data (only first 8 are significant)
(eg. your old password):>
WyoHish-Opt7 (Wyo-Hish-HYPHEN-Opt-SEVEN)
Ut4shlodWud_ (Ut-FOUR-shlod-Wud-UNDERSCORE)
1HiEbyen,Odd (ONE-Hi-Eb-yen-COMMA-Odd)
_1globAgevbu (UNDERSCORE-ONE-glob-Ag-ev-bu)
8grivLifNuj& (EIGHT-griv-Lif-Nuj-AMPERSAND)
ovHatsEw>Ob2 (ov-Hats-Ew-GREATER THAN-Ob-TWO)
```

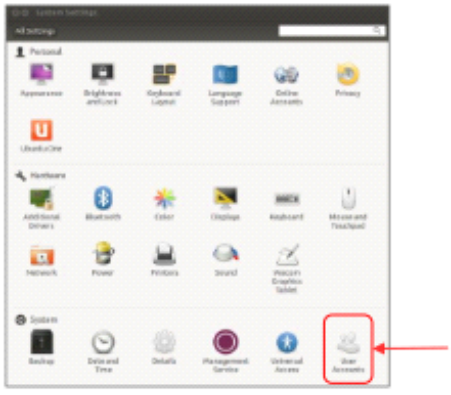
See: **/etc/passwd** for a list of user accounts.

Accounts

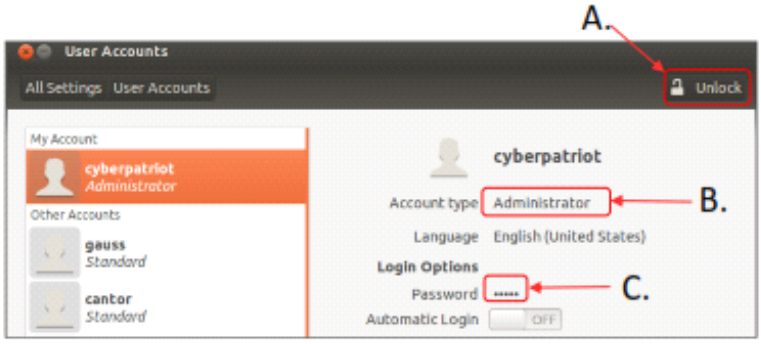
Disable unnecessary accounts

User Accounts

1.



2.



- Click User Accounts in the System Settings window
- As in Windows, it is important to restrict root (Admin) privileges and password protect all accounts
  - A. To make account management changes, you must enact root permissions by clicking Unlock and authenticate yourself by entering your password
  - B. Switch users from Administrator to Standard User by clicking next to Account Type
  - C. Change passwords by clicking the asterisks next to the Password option

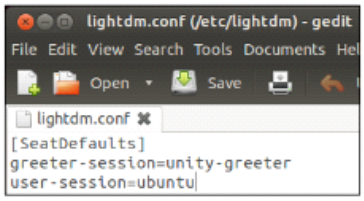
© Air Force Association 3

Using gedit to Turn off the Guest Account

- Like in Windows, the Ubuntu guest account is turned on by default
  - You should disable it so people can't access the computer anonymously
- The guest account is controlled by LightDM, the display manager controlling the Ubuntu login screen
- To turn off the guest account, edit the LightDM file:
  - After root authenticating, type `gedit /etc/lightdm/lightdm.conf`

```
root@ubuntu:/home/cyberpatriot# gedit /etc/lightdm/lightdm.conf
```

  - Add the line `allow-guest=false` to the end of the Light DM file that pops up and click Save
  - Restart your system and click your username button in the top-right corner of your desktop. The guest account should be disabled.



Sources: https://help.ubuntu.com/8.04/serverguide/C/user-management.html, http://askubuntu.com/questions/451526/removing-guest-session-at-login-in-ubuntu-14-04

© Air Force Association

You can also locate user accounts in **/etc/passwd**

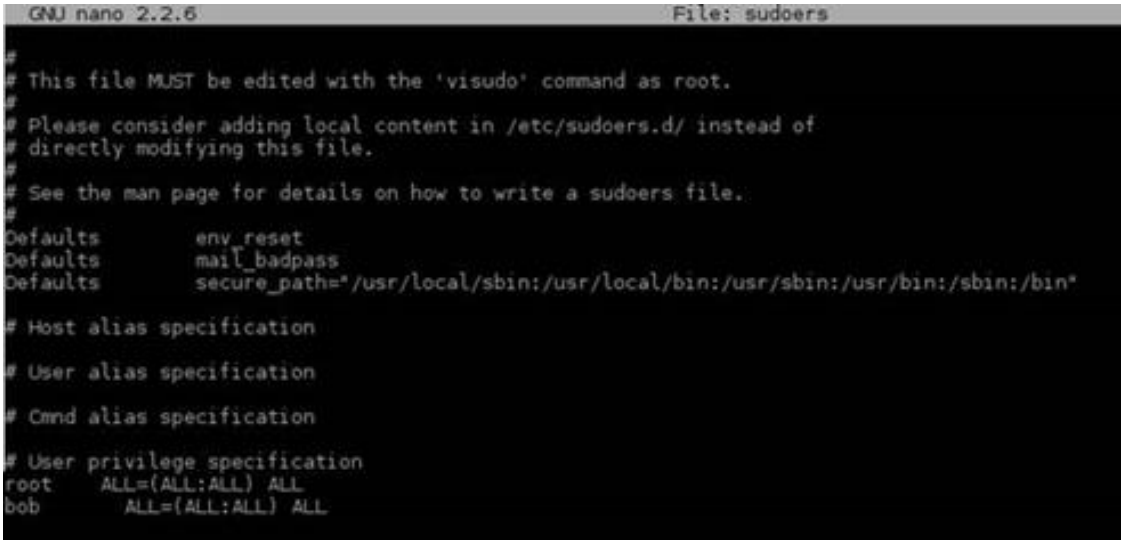
Make sure that there are no unauthorized accounts that have a user ID of 0.

Make sure that there are no unauthorized accounts in the *sudo* or *admin* group in **/etc/group**

Before doing anything else, make sure your user account (you should not be navigating around in *root*) is a part of the **/etc/sudoers** file!

sudo visudo /etc/sudoers

Find the line "# User privilege specification" and below this line add yourself using the following format:
username ALL=(ALL:ALL) ALL



```
GNU nano 2.2.6                    File: sudoers

#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults        env_reset
Defaults        mail_badpass
Defaults        secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL
bob       ALL=(ALL:ALL) ALL
```

Hit **Ctrl+O** to save the file, hitting **Enter** to confirm. The only users located in this section should be "root" and yourself, so if there are any other usernames under this section, remove them.

Now that we are sure our user account has proper priv escalation, we can confidently **LOCK OUT THE ROOT ACCOUNT.**

IF openssh-server is installed, disable *PermitRootLogin* in the *sshd_config* file.

sudo nano /etc/ssh/sshd_config

Find “PermitRootLogin” and switch “yes” to “no”

```
# Authentication:
LoginGraceTime 120
PermitRootLogin no
StrictModes yes
```

Ctrl+O and then **Enter** to save. Then use the service <name> <operation> command to restart (operation) ssh (name).

sudo service ssh restart

Next, we need to disable logging into the root account within the console itself. We do this by manipulating the password field in the shadow file. We can automate this task with various commands:

sudo passwd –l root **<---------- (this is an "l" as in {lock, lava, love…})**

or

sudo usermod –L root

or

sudo usermod –p ‘!’ root

Each method has its pros and cons. Locking the account with the *passwd* command or *usermod* **–L** switch prepends an “!” before the hashed password, useful for if you ever wish to unlock the account in the future. *Usermod* with the **–p** switch will substitute the hashed password in entirety with an exclamation point, completely locking the account out and not allowing you to revert.

To demonstrate how these commands work, here is how to manually disable the root account: first, **nano** into the **shadow** file

sudo nano /etc/shadow

Find the user “root” (usually first line), and in the second field, prepend an exclamation point to the hash. The following two images will present **a BEFORE and AFTER <u>respectively</u>**.

Do you notice the prepended **!** in the second image?
Now that we have disabled the root account after ensuring our priv escalation remains intact, we should lock out other unnecessary accounts.

First, look in the *passwd* file for any accounts that may be using a hashed password (any account with an “x” denoted in the second field).

sudo cat /etc/passwd

```
root@localhost:/etc# cat passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
```

What does this mean?

```
oracle:x:1021:1020:Oracle user:/data/network/oracle:/bin/bash
1      2 3    4        5                6              7
```

1. **Username** – What is used to login with.
2. **Password** – An **x** means that the password is encrypted in the /etc/shadow file
3. **User ID** – Users must have an identification number. The **root** user should have a UID of **0**; the range of [**1**,**99**] is reserved for general accounts, while the range of [**100**,**999**] is reserved for other administrative accounts and services.
4. **Group ID** – Users belong to groups, so you can give one group a special privilege without having to give potentially thousands of users (belonging to that one group) each that individual privilege.
5. **Comment** – this field does not affect the system, but is used for organizational purposes by the administrator to associate generic information with that user account. Remove any sensitive information you find from this field!
6. **Home Directory** – this is the absolute path for the home directory of the user (where user specific files (like music, documents, custom scripts) and configurations will go). For root, this is generally **/root** ; for other accounts, it is generally **/home/bob**
7. **Shell** – this defines the shell or custom command to use with the account after logging in. Basically, a shell is a command line program that takes your commands from the keyboard and passes them over to the operating system.

Tell your note taking partner the names of the accounts that might be using a password (**x** field). From there, exit out of *passwd* (**Ctrl+X**) and lock out the unnecessary accounts. This means, lock out every account ****EXCEPT** for your account!** We have already demonstrated how to lock out accounts in multiple ways. For review, use the *passwd* command with the appropriate switch and syntax to lockout the associated account.

sudo passwd –l sys

Next, nano into the *shadow* file and take note of any accounts that are using a hashed password. Note off the names of these accounts to your note taking partner and then Ctrl+X out of the interface. **If you installed *VMWare Tools for Linux***, you can make a backup of the shadow file and drag it to the Windows host, if your note taking partner finds that to be easier.

Now, head back to the *passwd* file and write down the GIDs (group ID) of each "special" account that was noted.
This should be the 4th field.

Once you have each of the special group IDs written down, navigate to the *group* file and take note of the strings associated with each ID number. For example, you will see the text "*root*" to the left of the number "*0*." *root* would be considered the string, whereas the 0 would be considered the group ID (an int data type). Your note taking partner should now have a list that looks like:

User Name	Group ID Number	Group Name
root	0	root
bob	27	sudo
...	...	...

Visudo back into the *sudoers* file:

sudo visudo /etc/sudoers

Now, instead of looking under *"# User privilege specification*," we want to look for groups under the *"# Allow members of group sudo to execute any command*" line. Generally, the group named "sudo," which might be associated with GID 27, is found under this line as so:

```
# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:

#includedir /etc/sudoers.d
```

Remove any group names here that match any of the special group names that were written down. If a malicious user exists in the *sudo* group, consider removing the %sudo line if your current account is not dependent upon it. Give special notice to the *#includedir* line. If this line is uncommented (meaning, the pound # symbol is not prepended), or there is another line within the file using the **includedir** directive, investigate that file and comment it out if it turns out to be malicious.

? *What users should be in the sudoers file?*

A: Essentially, only **authorized** users should be in the sudoers file. In the case of a competition, refer to the competition scenario that explicitly states what the usernames

What users should be in the sudoers file?

A: Essentially, only **authorized** users should be in the sudoers file. In the case of a competition, refer to the competition scenario that explicitly states what the usernames of all administrator accounts are. You use this information to lockdown your sudoers file. Based on this information, you have multiple options:

Case I) Make all of the authorized users a part of the same group (usually named "sudo"). Then, you can add all of those users to the sudoers file by using

%sudo ALL=(ALL:ALL) ALL

Case 2) For each individual authorized username, add it to the sudoers file using

username ALL=(ALL:ALL) ALL

Case 3) You have multiple files containing sudoers rules located in a directory /etc/sudoers.d -- then you can include all of those rules using

includedir /etc/sudoers.d

**NOTE that Case 3 is usually not necessary in the case of a competition and as such, you should actually be on the lookout for any pre-existing "includedir" directives which you will most likely want to delete!

The IDEAL sudoers file would be essentially blank, with only one line:

%sudo ALL=(ALL:ALL) ALL

where all authorized admins are part of the "sudo" group before making this change.

Finally, after you have **locked all unnecessary accounts and have checked for malicious groups within sudoers,** you should remove all of the “special accounts” that had hashed passwords, but were not accounts you had created.

sudo deluser maliciousbob

Tip: pay attention to the competition scenario! You might need to maintain certain user accounts rather than deleting them. In this case, you would want to **lock** them, nothing more.

Don't forget to **check for hidden user home directories**:

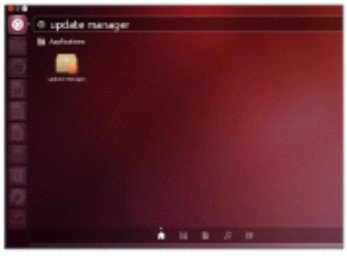
cd /home
sudo ls -Ra *

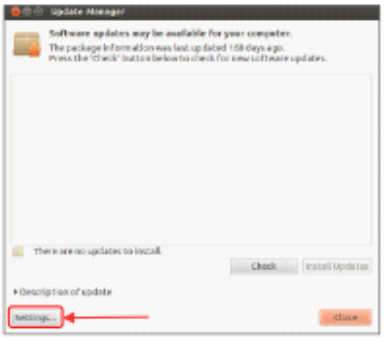
Updates

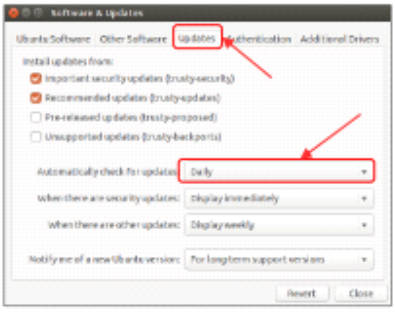
Installing and Automating Updates

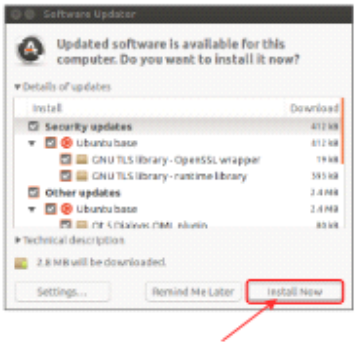
- The open-source community regularly develops improvements and patches for Ubuntu
- You should install these updates regularly

1. Click the Ubuntu button in the menu bar and search for Update Manager
2. Click Settings on the Update Manager Screen
3. To set automatic updates, go to the Updates Tab and make sure "Automatically check for updates" is set to "Daily"
4. After applying the changes, install any available updates from the main Update Manager window

1.


2.


3.


4.


© Air Force Association

4

Update the list of available packages

To do this, use the *update* option with apt-get.

sudo apt-get update

```
w@ubuntu:~$ sudo apt-get update
[sudo] password for w:
Hit http://us.archive.ubuntu.com precise Release.gpg
Get:1 http://extras.ubuntu.com precise Release.gpg [72 B]
Get:2 http://us.archive.ubuntu.com precise-updates Release.gpg [198 B]
Get:3 http://us.archive.ubuntu.com precise-backports Release.gpg [198 B]
Hit http://extras.ubuntu.com precise Release
Hit http://us.archive.ubuntu.com precise Release
Get:4 http://security.ubuntu.com precise-security Release.gpg [198 B]
Hit http://ppa.launchpad.net precise Release.gpg
97% [Waiting for headers] [Waiting for headers] [Waiting for headers] [Waiting
```

Upgrade to the latest versions of packages

To do this, use the *upgrade* option with apt-get.

sudo apt-get upgrade

```
w@ubuntu:~$ sudo apt-get upgrade
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages have been kept back:
  libminiupnpc8 linux-headers-generic-lts-raring
  linux-image-generic-lts-raring
The following packages will be upgraded:
```

Make sure that each critical service in your scenario is up to date!

Upgrade distribution (read scenario first)

If you are upgrading from an older version of Debian or from < Ubuntu 14:

sudo apt-get dist-upgrade

Otherwise:

sudo apt-get full-upgrade

To install software

sudo apt-get install softwareName1 softwareName2

To remove software

sudo apt-get remove softwareName1 softwareName2

To remove a service

sudo apt-get autoremove --purge serviceName

```
w@ubuntu:~$ sudo apt-get autoremove --purge apache2
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages will be REMOVED:
  apache2* apache2-mpm-worker* apache2-utils* apache2.2-common*
  thunderbird-globalmenu*
0 upgraded, 0 newly installed, 5 to remove and 167 not upgraded.
After this operation, 1,373 kB disk space will be freed.
Do you want to continue [Y/n]?
```

Firewall

Enabling the Firewall

- Enable the Ubuntu Built-in Firewall (UFW) to prevent unauthorized access to the computer
  - The UFW is deactivated by default
- By default, UFW is only accessible by command line
- You can download Gufw, a graphical firewall interface, from the Software Center and use it to make changes to the UFW in the GUI
  - You might need to install Ubuntu updates before installing Gufw

Source: https://help.ubuntu.com/community/UFW

© Air Force Association

5

Command to enable UFW: **sudo ufw enable**

Enable SYN cookie protection: **sysctl -n net.ipv4.tcp_syncookies**

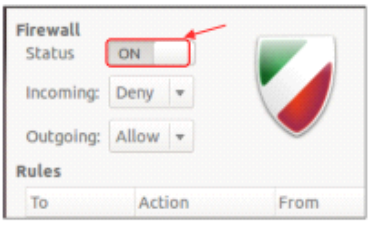
Deny all inbound traffic: **sudo ufw default deny incoming**

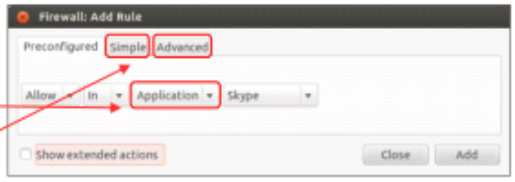
Allow all outbound traffic: **sudo ufw default allow outgoing**

For each critical service, allow inbound connections. Example: sudo ufw allow ssh
sudo ufw allow 22
sudo ufw allow http
sudo ufw allow 80
sudo ufw allow https
sudo ufw allow 443

Using Gufw

- After downloading Gufw from the Software Center, click the Ubuntu button in your menu bar → Search → Firewall Configuration
- Click the Unlock button on the Gufw window → Enact root permissions by authenticating → Turn Firewall Status On
- The default (and recommended rules) governing traffic are to Deny all incoming traffic and Allow all outgoing traffic
- The Reject option is the same as Deny, but also sends a notification to the sender that connection has been blocked
- The Preconfigured rule panel allows incoming and/or outgoing traffic to be controlled for certain applications or services
  - Similar to the Windows Firewall Exceptions list
  - Open entire ports by clicking the Simple or Advanced tabs





Source: https://help.ubuntu.com/community/Gufw

© Air Force Association

Chapter %00

Sunday, January 13, 2019 2:21 PM

AUTOMATE IT

Automating the Checklist

Sunday, January 13, 2019 2:22 PM

There are a collection of scripts that can be used to automatically complete various parts of this checklist. It is recommended that you launch these. It can potentially give you hours of extra time to look at other issues.

Before executing them, please read and make sure you understand these instructions so everything can go smoothly.

1. Before executing a script, open the script up in an editor and read the description. Some of the scripts have an area for you to configure it. These areas are marked by the comment, **#CHANGE THIS**. Once you understand what the script is intended to do and can configured it, you are ready to execute it.
2. When you execute the script, do so with administrative privileges. A lot of the scripts require permission to make changes to the system. The command to do this is:
   sudo sh nameofscript.sh

3. It is recommended that you launch the scripts in the following order:

cybersaxon
-scripts-m...

- **Policies**
  - **aging.sh** (sets password aging policies in logins.def)
  - **complexity.sh** (sets complexity requirements in common-password)
  - **lockout.sh** (sets lockout policy in common-auth)
- **Users**
  - Edit the **users.conf** file to add authorized users. Add one username per line
    - Default users that come with Ubuntu 16.04 have already been added
  - **unauthorizedfiner.sh** (This will create a file on the computer that shows a list of unauthorized users)
  - **rmunauthorized.sh** (This will delete unauthorized users on the computer)
  - **chpassword.sh** (This will change the password of all authorized users on the computer located in *users.conf*)
  - Edit the **lockusers.conf** file to add restricted users you would like to lock. Add one username per line
  - **userlocker.sh** (This will lock out user accounts that are defined in a list)
  - **zerofinder.sh** (This will find all users that have a UID of 0 (which means the user is root))
  - Edit the **groups.conf** file to add groups to monitor (the groups *sudo* and *admin* are added by default)
  - **groupfinder.sh** (This will find all users belonging to groups that are defined in *groups.conf*)
  - **disableguest.sh** (This will disable the guest account)
  - **disableautologin.sh** (This will disable autologin for all user accounts)
- **Permissions**
  - **showcron.sh** (this will show the crontab of all users)
  - **clearcron.sh** (this will reset the root crontab)
  - **limitcron.sh** (this will restrict cron so that only root can use it)
  - **findsevens.sh** (this will find files that have read,write,execute permissions)
- **Hardening**
  - **ape.sh** (This will clean up the apt sources for Ubuntu 16.04)
  - **update.sh** (This will run updates and upgrades)
  - **rc.sh** (This will clean up the rc file)
  - **ssh.sh** (Only execute this if SSH is a critical service! This will harden openssh-server)
  - **vsftpd.sh** (Only execute this if vsftpd is a critical service! This will harden vsftpd)
  - **apache2.sh** (Only execute this is apache2 is a critical service! This will harden apache2)
  - **mysql.sh** (Only execute this if mysql-server is a critical service! This will harden mysql-server)
  - **manualinspection.sh** (This will take you through key files that might need hardening.)
- **Networking**
  - **sysctl.sh** (This will harden sysctl)
  - **firewall.sh** (This will install ufw, enable it, and add rules)
  - **netinspector.sh** (This will inspect network connections)
  - **iptables.sh** (OPTIONAL! Only execute this if you need more advanced firewall rules!)
- **Malware - MAKE SURE YOU HAVE DONE FORENSICS QUESTIONS FIRST!**
  - **media.sh** (This will remove media files)
  - **rmexe.sh** (This will remove exe and msi files)
  - **apparmor.sh** (This will install apparmor and profiles)
  - **malwarepurge.sh** (This will delete common malware and hacking tools)

- **servicepurge.sh** (This will take you through an interactive wizard to delete services. Make sure you answer the questions very carefully so that you do not accidentally delete any critical services!)
- **chkrootkit.sh** (This will install chkrootkit and run it with recommended options)
- **rkhunter.sh** (This will install rkhunter and run it with recommended options)
- **clamav.sh** (This will install clamav, update it, and scan the system for viruses)
- **lynis.sh** (This will install lynis and run it with recommended options)
- **findphp.sh** (This will display a list of php files on the system. This is useful for checking for possible PHP backdoors)

Chapter 3. SECPOL && INSECURE DEFAULTS

Install cracklib

Security policies are definitions that define the constraints of the security mechanism(s) itself. Assuming the security system is properly implemented; policies will play an important role. Requiring a password is an important mechanism for most security models. Requiring a minimum password length, via policies, makes the model even stronger.

****YOU CAN AUTOMATE THIS PART FROM CHAPTER %00**

First, install the *libpam_cracklib* module, which is important for managing the rules and roles of passwords.

sudo apt-get install libpam-cracklib

Tip: Append –y to the apt-get install x command to auto install without it asking the user.

Test out cracklib after it has been installed.

```
cyberpatriot@ubuntu:~$ apg

Please enter some random data (only first 8 are significant)
(eg. your old password):>
Nicyehons0 (Nic-ye-hons-ZERO)
yievesGoab0 (yiev-es-Goab-ZERO)
vipFadIs2 (vip-Fad-Is-TWO)
CeishVinkAj1 (Ceish-Vink-Aj-ONE)
Ic4dacks (Ic-FOUR-dacks)
Jiasbucub9 (Jias-buc-ub-NINE)
cyberpatriot@ubuntu:~$ echo "abc"
abc
cyberpatriot@ubuntu:~$ echo "abc" | cracklib-check
abc: it is WAY too short
cyberpatriot@ubuntu:~$ echo "password" | cracklib-check
password: it is based on a dictionary word
cyberpatriot@ubuntu:~$ echo "Nicyehons0" | cracklib-check
Nicyehons0: OK
cyberpatriot@ubuntu:~$ echo "Ic4dacks" | cracklib-check
Ic4dacks: OK
```

Set minimum password length

****YOU CAN AUTOMATE THIS PART FROM CHAPTER %00**

sudo nano /etc/pam.d/common-password

Find the line that begins with *password requisite* and make sure it looks like the following (all in one line):

password requisite pam_cracklib.so retry=3 minlen=8 difok=3 reject_username minclass=3 maxrepeat=2 dcredit=1 lcredit=1 ocredit=1

retry: the amount of password attempts allowed

minlen: minimum length of passwords

difok: minimum requirement of different characters between passwords

reject_username: does not allow the use of the username for the password

minclass: minimum class of different characters required (e.g. capitalized, symbol, number)

maxrepeat: maximum amount of repeatable characters allowed (e.g. the letter “a” allowed to be used three times)

dcredit: the minimum amount of digits required in the password

lcredit: the minimum amount of lower case letters required in the password

ocredit: the minimum amount of other symbols that are to be required in the password

Next, find the line that has **pam_unix.so** in it and add **minlen=8** *to the end of it.* Re-read the previous sentence to make sure you understand it.

Enforce account lockout policy

****YOU CAN AUTOMATE THIS PART FROM CHAPTER %00**

sudo nano /etc/pam.d/common-auth

Find the line with *pam_tally2.so* in it and add the following to the end: **deny=5 unlock_time=1800**
Re-read the previous sentence once more to make sure that you understand.

Enable password history

Inside of **/etc/pam.d/common-password** , enable password history by adding the following line just beneath the pam_cracklib.so line:

password requisite pam_pwhistory.so use_authtok remember=24 enforce_for_root

After you make this edit, in the same file, find the line dealing with *password pam_unix.so* and make sure it looks like:

password [success=1 default=ignore] pam_unix.so obscure use_authtok try_first_pass sha512

The file might look similar to this:

```
# here are the per-package modules (the "Primary" block)
password        requisite                       pam_cracklib.so retry=3 minlen=8 difok=3 reject_username minclass=3 maxrepeat=2 dcre
password        requisite                       pam_pwhistory.so use_authtok remember=24 enforce_for_root
password        [success=1 default=ignore]      pam_unix.so obscure use_authtok try_first_pass sha512 rounds=20000
```

Then, to finish enabling password history:

Navigate to **/etc/security** and enter the following three commands to create and secure the *opasswd* file:

touch /etc/security/opasswd

chown root:root /etc/security/opasswd

chmod 600 /etc/security/opasswd

Enforce password expiration

****YOU CAN AUTOMATE THIS PART FROM CHAPTER %00**

sudo nano /etc/login.defs and make the following changes:

```
#
# Password aging controls:
#
#       PASS_MAX_DAYS   Maximum number of days a password may be used.
#       PASS_MIN_DAYS   Minimum number of days allowed between password changes.
#       PASS_WARN_AGE   Number of days warning given before a password expires.
#
PASS_MAX_DAYS   90
PASS_MIN_DAYS   7
PASS_WARN_AGE   14
```

THEN, *for each* user account located in **/etc/passwd**, use the **chage** command. For example:

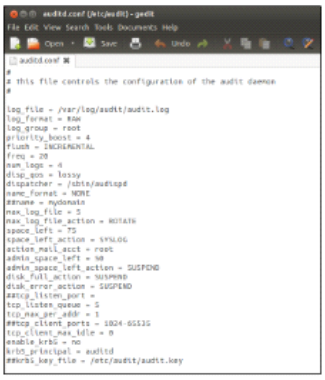
sudo chage -m 7 -M 90 -W 14 root

Enable auditing

Setting Audit Policies

- Unlike Windows, auditing is not set up by default in Ubuntu
- Three step process to setting up audits:
  1. Install the auditing program by typing `apt-get install auditd`
  2. Enable audits by typing `auditctl -e 1`
  3. View and modify policies by typing `gedit /etc/audit/auditd.conf`

3.



2.

```
root@ubuntu:/home/cyberpatriot# auditctl -e 1
AUDIT_STATUS: enabled=1 flag=1 pid=4229 rate_limit=0 backlog_limit=320 lost=50 b
acklog=0
```

© Air Force Association

Insecure Defaults

Ensure console security

An automated task or remote command could enter in *Ctrl+Alt+Del*, rebooting the linux image. This has the potential to damage hardware and or software depending on the circumstance.

Edit the file */etc/init/control-alt-delete.conf*

sudo nano /etc/init/control-alt-delete.conf

Comment out the line *exec shutdown –r now "Control+Alt+Delete pressed"*

```
# control-alt-delete - emergency keypress handling
#
# This task is run whenever the Control-Alt-Delete key combination is
# pressed, and performs a safe reboot of the machine.

description     "emergency keypress handling"
author          "Scott James Remnant <scott@netsplit.com>"

start on control-alt-delete

task
exec shutdown -r now "Control-Alt-Delete pressed"
```

To comment out a line, prepend a pound # symbol to the line.

```
task
#exec shutdown -r now "Control-Alt-Delete pressed"
```

Save the file.

Restrict *su* to admin group

sudo dpkg-statoverride --update --add root admin 4750 /bin/su

Disable IPv6 (if not needed)

****YOU CAN AUTOMATE THIS PART FROM CHAPTER %00**

To disable IPv6, create a new file called ***aliases*** in the */etc/modprobe.d* directory

sudo nano /etc/modprobe.d/aliases

```
  GNU nano 2.2.6          File: /etc/modprobe.d/aliases

alias net-pf-10 off
```

Save the file and reboot the machine to reflect the changes.

Secure shared memory

By default, shared memory is mounted to read, write, and execute programs. This default setting might lead to some vulnerabilities in specific circumstances. To account for this, add the following line to your */etc/fstab* file.

tmpfs /dev/shm tmpfs defaults,ro 0 0

```
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
proc            /proc           proc    nodev,noexec,nosuid 0       0
# / was on /dev/sda1 during installation
UUID=0e8f3f31-80f2-4a40-956d-fe9e44ebacb1 /               ext4    errors=remount-ro 0       1
# swap was on /dev/sda5 during installation
UUID=a91860b5-13b0-449b-aa18-fc8175d8e6d2 none            swap    sw              0       0
/dev/fd0        /media/floppy0  auto    rw,user,noauto,exec,utf8 0       0

#added line VVVVVVVVVVV

tmpfs     /dev/shm     tmpfs     defaults,ro     0     0
```

Add this line pl0x

Restrict DMESG

To restrict non-administrative groups from accessing DMESG, set *dmesg_restrict* to **1** with the following command:

sudo sh -c 'echo 1 > /proc/sys/kernel/dmesg_restrict'

```
w@ubuntu:~$ sudo sh -c 'echo 1 > /proc/sys/kernel/dmesg_restrict'
w@ubuntu:~$ dmesg
dmesg: klogctl failed: Operation not permitted
```

Secure the Home directory

sudo chmod 0700 /home/username

Be sure to replace "username" with your username. You should do this step against all authorized user accounts.

For example: Authorized Users = {Bob, Joe, Rachel}

sudo chmod 0700 /home/Bob

sudo chmod 0700 /home/Joe

sudo chmod 0700 /home/Rachel

Lock all accounts with empty passwords

First, enter in the following command:

awk -F: '($2 == "") {print}' /etc/shadow

Then, lock all of the accounts that are printed using **passwd -l accountName** **<---------- (l as in {L for LOCK})**

Make Sure No Non-Root Accounts UID == 0

****YOU CAN AUTOMATE THIS PART FROM CHAPTER %00**

Only the "root" account should have a UID of 0. Therefore, if any other account has a UID of 0, they should not.
First, type the following command to search for all users that have UID == 0:

awk -F: '($3 == "0") {print}' /etc/passwd

If there are users listed other than "root," delete the accounts or assign them a unique UID > 1000.

Harden Linux Kernel Parameters

****YOU CAN AUTOMATE THIS PART FROM CHAPTER %00**

Your **/etc/sysctl.conf** file have the following:

```
# Turn on execshield
kernel.exec-shield=1
kernel.randomize_va_space=1
# Enable IP spoofing protection
net.ipv4.conf.all.rp_filter=1
# Disable IP source routing
net.ipv4.conf.all.accept_source_route=0
# Ignoring broadcasts request
net.ipv4.icmp_echo_ignore_broadcasts=1
net.ipv4.icmp_ignore_bogus_error_messages=1
# Make sure spoofed packets get logged
net.ipv4.conf.all.log_martians = 1
```

⭐ Disable unwanted SUID and SGID bins

SUID/SGID enabled files can be executed by all local and remote users. Find all of these files and investigate them as necessary using the following two commands:

find / -perm +4000

find / -perm +2000

Chapter 4. HARDENING

Patch Vulnerabilities

Patch Heartbleed

Heartbleed is a serious vulnerability in OpenSSL software that allows sensitive information to be stolen. The attacker can eavesdrop on traffic by compromising the secret keys that were used to encrypt the data.

To see if you are vulnerable to Heartbleed, run the following command:
openssl version

If you are not running a version >= 1.0.1g , you may be vulnerable. Heartbleed targets OpenSSL 1.0.1 through 1.0.1f (inclusive)

If you are vulnerable, run the following command:
sudo apt-get update && sudo apt-get upgrade openssl

If you are vulnerable and the above command does not patch openssl, do the following:

1. *Download the latest version of OpenSSL.*
   wget http://www.openssl.org/source/openssl-1.0.1g.tar.gz

2. *Extract the archive.*
   tar -zxvf openssl-1.0.1g.tar.gz

3. *Run the config script.*
   ./config --prefix=/usr --openssldir=/usr/local/openssl shared

4. *Compile openssl.*
   make
   make test
   make install

Patch Shellshock/Bashdoor

Shellshock is a vulnerability in Bash shell that allows for remote code execution. An attacker can execute arbitrary commands to infiltrate the system.

To see if you are vulnerable to Shellshock, also known as Bashdoor, run the following command:

wget https://shellshocker.net/shellshock_test.sh -O - | sh

If you are vulnerable, run the following command:

wget https://shellshocker.net/fixbash -O - | sh

Patch POODLE

POODLE is a vulnerability that targets SSL 3.0, allowing a MiTM attacker to decrypt "secure" traffic.
The solution is to disable SSL 3.0 completely.

Following the instructions found in the "Ensure secure SSL configuration" step of the "Harden Services" subsection in Chapter 4 should solve this problem.

Mandatory Access Control

⭐ Enable apparmor

****YOU CAN AUTOMATE THIS PART FROM CHAPTER %00**

sudo apt-get install apparmor

Harden Services

Harden mysql

****YOU CAN AUTOMATE THIS PART FROM CHAPTER %00**

<u>This step is only necessary if apache2 and or mysql is/are [a] critical service(s) to maintain as required by your scenario.</u>

sudo mysql_secure_installation

```
w@ubuntu:~$ sudo mysql_secure_installation
```

```
Change the root password? [Y/n] y
New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables..
 ... Success!
```

```
Remove anonymous users? [Y/n] y
 ... Success!

Normally, root should only be allowed to connect from 'localhost'.  This
ensures that someone cannot guess at the root password from the network.
```

```
Disallow root login remotely? [Y/n] y
 ... Success!
```

```
Remove test database and access to it? [Y/n] y
 - Dropping test database...
```

```
Reload privilege tables now? [Y/n] y
 ... Success!

Cleaning up...
```

sudo service mysql restart

Harden SSH

****YOU CAN AUTOMATE THIS PART FROM CHAPTER %00**

Open up the **sshd_config** file located in */etc/ssh*

sudo nano /etc/ssh/sshd_config

First, set **PermitRootLogin** to **no**
Then, set **LoginGraceTime** to **30**

```
# Authentication:
LoginGraceTime 30
PermitRootLogin no
StrictModes yes
```

Make sure **RSAAuthentication** is set to **yes**
PubkeyAuthentication is set to **yes**

```
RSAAuthentication yes
PubkeyAuthentication yes
#AuthorizedKeysFile     %h/.ssh/authorized_keys
```

Make sure **PermitEmptyPasswords** is set to **no**
PasswordAuthentication should also be set to **no**

```
# To enable empty passwords, change to yes (NOT RECOMMENDED)
PermitEmptyPasswords no

# Change to yes to enable challenge-response passwords (beware issues with
# some PAM modules and threads)
ChallengeResponseAuthentication no

# Change to no to disable tunnelled clear text passwords
PasswordAuthentication no
```

You should also use SFTP instead of FTP, and also restrict SSH to an allowed set of users. To use SFTP instead of FTP, find the line

```
Subsystem sftp /usr/lib/openssh/sftp-server
```

And change it to

```
Subsystem sftp internal-sftp
```

To restrict SSH to a whitelist of users, use the **AllowUsers** directive. For example:

```
#custom off the chain stuff

AllowUsers user1 user2 user3 Bob tom
```

Finally, save the *sshd_config* file and restart the ssh service.

sudo service ssh restart

Harden apache

First, install *suhosin*, *mod-security*, and *mod-evasive*.

```
w@ubuntu:~$ sudo apt-get install php5-suhosin libapache-mod-security libapache2-mod-evasive
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  thunderbird-globalmenu
Use 'apt-get autoremove' to remove them.
The following extra packages will be installed:
  libapache2-modsecurity modsecurity-crs php5-common php5-fpm
Suggested packages:
  lua php-pear
The following NEW packages will be installed:
  libapache-mod-security libapache2-mod-evasive libapache2-modsecurity modsecurity-crs php5-common php5-fpm php5-suhosin
0 upgraded, 7 newly installed, 0 to remove and 167 not upgraded.
Need to get 3,804 kB/3,820 kB of archives.
After this operation, 11.4 MB of additional disk space will be used.
Do you want to continue [Y/n]?
```

Then, minimize your apache fingerprint by adding the following lines to the ***apache2.conf*** file located in */etc/apache2*:

```
#add this stuff fo sho

ServerSignature Off
ServerTokens ProductOnly
TraceEnable Off
```

Then, disable the ***autoindex*** and ***dav*** modules and enable the ***mod-security*** and ***mod-evasive*** modules.

sudo a2dismod autoindex sudo a2dismod dav

```
w@ubuntu:/etc/apache2$ sudo a2dismod autoindex
Module autoindex disabled.
To activate the new configuration, you need to run:
  service apache2 restart
```

sudo a2enmod mod-security

sudo a2enmod mod-evasive

Finally, rename *modsecurity.conf-recommended* to *modsecurity.conf* and set the **SecRuleEngine** directive to **On**.

cd /etc/modsecurity

sudo mv modsecurity.conf-recommended modsecurity.conf

```
  GNU nano 2.2.6                          File: modsecurity.conf

# -- Rule engine initialization ----------------------------------------------

# Enable ModSecurity, attaching it to every transaction. Use detection
# only to start with, because that minimises the chances of post-installation
# disruption.
#
SecRuleEngine On
```

Restart the apache2 service to reflect the changes.

sudo service apache2 restart

Secure php

IF you must maintain a webserver (*apache2* OR *nginx* OR *lighttpd*)

For steps 1, 2, 3, 4, 5, 7, 8:
sudo nano php.ini

Your *php.ini* file can be located by entering the following command:
php --ini

```
cyber@ubuntu: ~
cyber@ubuntu:~$ php --ini
Configuration File (php.ini) Path: /etc/php5/cli
Loaded Configuration File:         /etc/php5/cli/php.ini
Scan for additional .ini files in: /etc/php5/cli/conf.d
Additional .ini files parsed:      /etc/php5/cli/conf.d/05-opcache.ini,
/etc/php5/cli/conf.d/10-pdo.ini,
/etc/php5/cli/conf.d/20-json.ini,
/etc/php5/cli/conf.d/20-readline.ini
```

1. Enable PHP safe mode.
   Find the *safe_mode* directive in *your php.ini* file and set it to = *On*

   safe_mode = On

```
; Safe Mode
; http://php.net/safe-mode
safe_mode = On
```

2. Disable dangerous PHP functions.
   Find the *disable_functions* directive or create it **IF** it does not exist. Add the following [potentially dangerous] functions to disable:

   disable_functions = "file_get_contents, exec, passthru, system, shell_exec, popen, proc_open, pcntl_exec, eval, assert, fopen, bzopen, gzopen"

```
; This directive allows you to disable certain functions for security reasons.
; It receives a comma-delimited list of function names. This directive is
; *NOT* affected by whether Safe Mode is turned On or Off.
; http://php.net/disable-functions
disable_functions = "file_get_contents, exec, passthru, system, shell_exec, pop
```

3. Restrict PHP information linkage.
   Sometimes php becomes vulnerable to attacks. If you are running a vulnerable version of php, an attacker could find out that you are vulnerable to this attack through some basic recon. To restrict PHP information leakage, set the *expose_php* directive to *Off*.

   expose_php = Off

```
; Decides whether PHP may expose the fact that it is installed on the server
; (e.g. by adding its signature to the Web server header).  It is no security
; threat in any way, but it makes it possible to determine whether you use PHP
; on your server or not.
; http://php.net/expose-php
expose_php = Off
```

   To disable the display of PHP errors to visitors, set the *display_errors* directive to *Off*.

   display_errors = Off

```
; having the errors sent to STDOUT.
; Possible Values:
;   Off = Do not display any errors
;   stderr = Display errors to STDERR (affects only CGI/CLI binaries!)
;   On or stdout = Display errors to STDOUT
; Default Value: On
; Development Value: On
; Production Value: Off
; http://php.net/display-errors
display_errors = Off
```

4. Log all PHP errors.
   To log all PHP errors, set the *log_errors* directive to *On* and set the *error_log* directive.

   log_errors = On
   error_log = /var/log/httpd/php.log

```
; directive found below. While errors should not be displayed on productions
; servers they should still be monitored and logging is a great way to do that.
; Default Value: Off
; Development Value: On
; Production Value: On
; http://php.net/log-errors
log_errors = On
error_log = /var/log/httpd/php.log
```

5. Disallow file uploads.
   Unless your site is supposed to allow file uploads, HTTP file uploads should be disabled. To do this, set the *file_uploads* directive to *Off* .

   file_uploads = Off

```
;;;;;;;;;;;;;;;;;
; File Uploads ;
;;;;;;;;;;;;;;;;;

; Whether to allow HTTP file uploads.
; http://php.net/file-uploads
file_uploads = Off
```

6. Enable SQL safe mode.
   To ignore potentially dangerous arguments passed to mysql_connect() and mysql_pconnect() , set the *sql.safe_mode* directive to *On* .

sql.safe_mode = On

```
[SQL]
; http://php.net/sql.safe-mode
sql.safe_mode = On
```

7. Mitigate DoS attacks.
   To control resources and mitigate DoS attacks, change or add the *max_execution_time* , *max_input_time* , and *memory_limit* directives to:

   max_execution_time = 30
   max_input_time = 30
   memory_limit = 25M

```
; Maximum execution time of each script, in seconds
; http://php.net/max-execution-time
; Note: This directive is hardcoded to 0 for the CLI SAPI
max_execution_time = 30
```

```
; Maximum amount of time each script may spend parsing request data. It's a goo
; idea to limit this time on productions servers in order to eliminate unexpect
; long running scripts.
; Note: This directive is hardcoded to -1 for the CLI SAPI
; Default Value: -1 (Unlimited)
; Development Value: 60 (60 seconds)
; Production Value: 60 (60 seconds)
; http://php.net/max-input-time
max_input_time = 30
```

```
; Maximum amount of memory a script may consume (128MB)
; http://php.net/memory-limit
memory_limit = 25M
```

8. Install and enable suhosin.
   Suhosin is an advanced PHP security extension that protects against low-level attacks. To install, first add the Suhosin repos to *etc/apt/sources.list*

```
## /etc/apt/sources.list for Debian wheezy (amd64, i386, armhf)
deb http://repo.suhosin.org/ debian-wheezy main

## /etc/apt/sources.list for Ubuntu trusty (amd64 only)
deb http://repo.suhosin.org/ ubuntu-trusty main
```

Then, download and add the Suhosin archive key to apt.

wget https://sektioneins.de/files/repository.asc
sudo apt-key add repository.asc

```
cyber@ubuntu:~$ wget https://sektioneins.de/files/repository.asc
--2014-12-05 22:36:23--  https://sektioneins.de/files/repository.asc
Resolving sektioneins.de (sektioneins.de)... 85.214.50.221
Connecting to sektioneins.de (sektioneins.de)|85.214.50.221|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3210 (3.1K) [text/plain]
Saving to: ‘repository.asc’

100%[=====================================>] 3,210       --.-K/s   in 0.004s

2014-12-05 22:36:24 (845 KB/s) - ‘repository.asc’ saved [3210/3210]

cyber@ubuntu:~$ sudo apt-key add repository.asc
OK
```

From there, run **sudo apt-get update** and then install *php5-suhosin-extension* .

sudo apt-get update
sudo apt-get install php5-suhosin-extension

```
cyber@ubuntu:~$ sudo apt-get update
Ign http://us.archive.ubuntu.com utopic InRelease
Ign http://extras.ubuntu.com utopic InRelease
Ign http://us.archive.ubuntu.com utopic-updates InRelease
Ign http://us.archive.ubuntu.com utopic-backports InRelease
Hit http://extras.ubuntu.com utopic Release.gpg
```

```
cyber@ubuntu:~$ sudo apt-get install php5-suhosin-extension
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  php5-suhosin-extension
0 upgraded, 1 newly installed, 0 to remove and 57 not upgraded.
Need to get 290 kB of archives.
After this operation, 0 B of additional disk space will be used.
Get:1 http://repo.suhosin.org/ debian-wheezy/main php5-suhosin-extension i386 0
9.37-1 [290 kB]
Fetched 290 kB in 1s (154 kB/s)
Selecting previously unselected package php5-suhosin-extension.
(Reading database ... 169758 files and directories currently installed.)
Preparing to unpack .../php5-suhosin-extension_0.9.37-1_i386.deb ...
Unpacking php5-suhosin-extension (0.9.37-1) ...
```

Remove unnecessary files in /var/www

Search for and remove PHP shells *(locate "*.php*")*. Look for files that include filenames from the shell list below. Scripts that are encoded in base64 are also more than likely backdoors.

List of common PHP shells in alphabetical order:

accept_language PHP
Ajax_PHP Command Shell PHP
AK-74 Security Team Web Shell Beta Version PHP
Antichat Shell PHP
Antichat Shell v1.3 PHP
Antichat Shell v1.3 PHP
Antichat Shell. Modified by Go0o$E PHP
Ayyildiz Tim -AYT- Shell v 2.1 Biz PHP
Ayyildiz Tim -AYT- Shell v 2.1 Biz PHP
aZRaiLPhp v1.0 PHP
aZRaiLPhp v1.0 PHP
b374k PHP
b374k-mini-shell PHP
backupsql PHP
c0derz shell [csh] v. 0.1.1 release PHP
C2007Shell v. 1.0 pre-release build #16 Modded by Adora & u9 h4c93r PHP
c99 PHP
c99_locus7s PHP
c99_madnet PHP
c99_PSych0 PHP
c99_w4cking PHP
C99madShell v. 2.0 madnet edition PHP
C99Shell v. 1.0 beta (5.02.2005) PHP

C99Shell v. 1.0 pre-release build(safe-mode) PHP
CasuS 1.5 PHP
CmdAsp.asp.txt ASP
Crystal PHP
Crystal shell PHP
CrystalShell v.1 PHP
CTT Shell PHP
ctt_sh PHP
Cyber Shell PHP
Cyber Shell (v 1.0) PHP
cybershell PHP
CyberSpy5.Asp.txt ASP
dC3 Security Crew Shell PRiV PHP
dC3 Security Crew Shell PRiV PHP
Dive Shell 1.0 - Emperor Hacking Team PHP
Dive Shell 1.0 - Emperor Hacking Team PHP
DTool Pro PHP
Dx PHP
DxShell v1.0 PHP
DxShell.1.0 PHP
EFSO_2.txt ASP
ELMALISEKER Backd00r ASP
Elmaliseker.txt ASP
Gamma Web Shell Perl-Cgi
GFS Web-Shell PHP
GFS web-shell ver 3.1.7 - PRiV8 PHP
GFS web-shell ver 3.1.7 - PRiV8 PHP
gfs_sh PHP
go-shell Perl-Cgi
GRP WebShell 2.0 release build 2018 (C)2006,Great PHP
h4ntu shell [powered by tsoi] PHP
h4ntu shell [powered by tsoi] PHP
hiddens shell v1 PHP
iMHaBiRLiGi Php FTP PHP
iMHaPFtp PHP
indexer.asp.txt ASP
ironshell PHP
JspWebshell 1.2 PHP
JspWebshell 1.2 JSP
KA_uShell 0.1.6 PHP
KAdot Universal Shell v0.1.6 PHP
KAdot Universal Shell v0.1.6 PHP
klasvayv.asp.txt ASP
lamashell PHP
Liz0ziM Private Safe Mode Command Execuriton Bypass Exploit PHP
Liz0ziM Private Safe Mode Command Execuriton Bypass Exploit PHP
load_shell PHP
Loaderz WEB Shell PHP
Macker's Private PHPShell PHP

matamu PHP
Moroccan Spamers Ma-EditioN By GhOsT PHP
MyShell PHP
myshell PHP
mysql PHP
Mysql interface v1.0 PHP
Mysql interface v1.0 PHP
MySQL Web Interface Version 0.8 PHP
MySQL Web Interface Version 0.8 PHP
mysql_tool PHP
NCC-Shell PHP
NetworkFileManagerPHP PHP
NFM 1.8 PHP
NGH PHP
NIX REMOTE WEB SHELL PHP
NIX REMOTE WEB-SHELL v.0.5 alpha Lite Public Version PHP
NIX REMOTE WEB-SHELL v.0.5 alpha Lite Public Version PHP
nshell PHP
nstview PHP
nsTView v2.1 PHP
NTDaddy v1.9 ASP
Perl Web Shell by RST-GHC PL
PH Vayv PHP
PHANTASMA PHP
PHP Shell PHP
php-backdoor PHP
php-include-w-shell PHP
pHpINJ PHP
PHPJackal PHP
phpRemoteView PHP
PHPRemoteView PHP
PhpSpy Ver 2006 Perl-Cgi
PHVayv PHP
Predator PHP
Private-i3lue PHP
Private-i3lue PHP
pws PHP
r57 PHP
r57_iFX PHP
r57_kartal PHP
r57_Mohajer22 PHP
r57shell PHP
reader.asp.txt ASP
RedhatC99 [login=redhat-pass=root] PHP
RemExp.asp.txt ASP
rootshell PHP
Rootshell.v.1.0 PHP
ru24_post_sh PHP
s72 Shell v1.1 Coding PHP

s72 Shell v1.1 Coding PHP
Safe_Mode Bypass PHP 4.4.2 and PHP 5.1.2 PHP
Safe_Mode Bypass PHP 4.4.2 and PHP 5.1.2 PHP
Safe0ver Shell -Safe Mod Bypass By Evilc0der PHP
Safe0ver Shell -Safe Mod Bypass By Evilc0der PHP
SimAttacker - Vrsion 1.0.0 - priv8 4 My friend PHP
SimAttacker - Vrsion 1.0.0 - priv8 4 My friend PHP
simple_cmd PHP
simple-backdoor PHP
SimShell 1.0 - Simorgh Security MGZ PHP
SimShell 1.0 - Simorgh Security MGZs PHP
Sincap 1.0 PHP
Small Web Shell by ZaCo PHP
SnIpEr_SA Shell PHP
STNC WebShell v0.8 PHP
Uploader PHP
Web-shell (c)ShAnKaR PHP
WinX Shell PHP
WinX Shell PHP
Worse Linux Shell PHP
Worse Linux Shell PHP
zacosmall PHP
zehir4.asp.txt ASP
ZyklonShell PHP

Ensure secure SSL configuration

If your apache server is using SSL, then it is important that you use a secure configuration. If you are not using SSL, do not bother trying to apply these configurations.

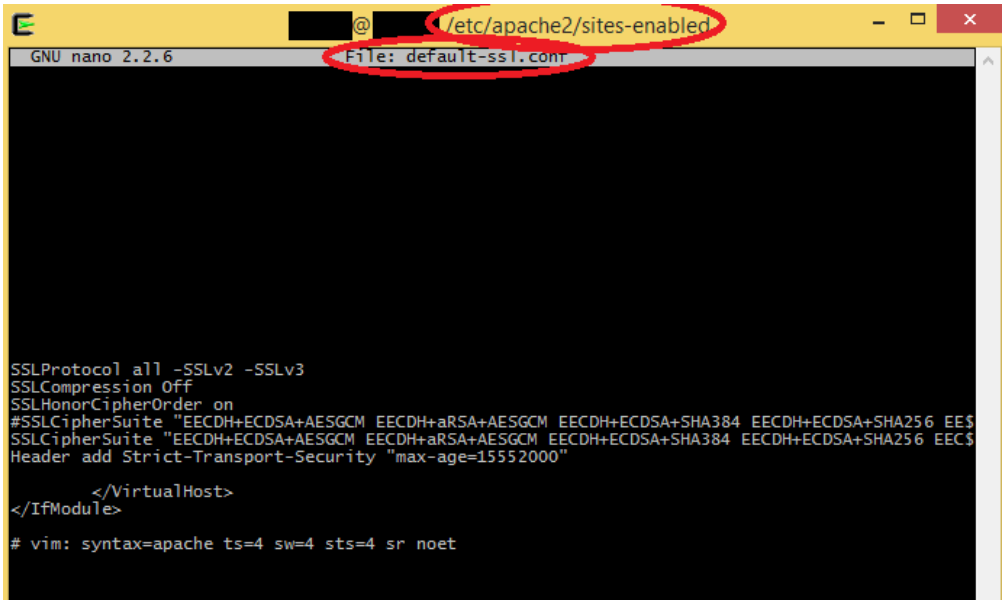
Navigate to your website's ssl conf file, which is normally located in the "**sites-enabled**" or "**sites-available**" directory under **/etc/apache2** IF you are using virtual hosts. Otherwise, revert to your apache conf file, i.e. **/etc/apache2/apache2.conf**

Add the following lines to your ssl server configuration file, i.e. **/etc/apache2/sites-enabled/default-ssl.conf** :

```
SSLProtocol all -SSLv2 -SSLv3
SSLCompression Off
SSLHonorCipherOrder on

SSLCipherSuite "EECDH+ECDSA+AESGCM EECDH+aRSA+AESGCM EECDH+ECDSA+SHA384 EECDH+ECDSA+SHA256 EECDH+aRSA+SHA384 EECDH+aRSA+SHA256 EECDH+aRSA+RC4 EECDH EDH+aRSA RC4 !aNULL !eNULL !LOW !3DES !MD5 !EXP !PSK !SRP !DSS +RC4 RC4"

Header add Strict-Transport-Security "max-age=15552000"
```



But what does it all mean?

SSLProtocol all -SSLv2 -SSLv3 : SSLv2 and SSLv3, considered obsolete, are disabled.
SSLCompression Off : Disable compression to mitigate the CRIME vulnerability (Compression Ratio Info-Leak Made Easy)
SSLHonorCipherOrder On : Honor which cipher order a browser must use to mitigate the BEAST vulnerability (prefer cipher suites)
SSLCipherSuite "EECDH+ECD.. : The order of ciphers, preferring AESGCM (TLS 1.2) when possible
Header add Strict-Transport-Security "max-age-15552000" : Enforces HTTPS, no HTTP allowed

Before you restart apache2 to apply the changes, enable **mod_headers**. This will allow for the HSTS setting reflected above.

sudo a2enmod mod_headers
sudo service apache2 restart

Install Bastille

First, install Bastille.

sudo apt-get install bastille

Then you will want to run bastille with administrative privileges.

Command line: **sudo bastille**
GUI: **sudo bastille -x**

If you want to GUI and it will not load, you need to install Perl's Tk module: **sudo apt-get install perl-tk**

Bastille will ask you a set of questions. Answer as follows:

Q: Would you like to set more restrictive permissions on admin utilities?
A: Yes

Q: Would you like to disable SUID status for mount/umount?
A: Yes

Q: Would you like to disable SUID status for ping?
A: Yes

Q: Would you like to disable SUID status for traceroute?
A: Yes

Q: Should Bastille disable clear-text r-protocols that use IP-based authentication?
A: Yes

Q: Would you like to enforce password aging?
A: Yes

Q: Do you want to set the default umask?
A: Yes (use "077" (no quotation marks))

Q: Should we disallow root login on tty's 1-6?
A: Yes

Q: Would you like to password-protect the GRUB prompt?
A: Yes

Q: Would you like to set a default-deny on TCP Wrappers and xinetd?
A: No

Q: Should Bastille ensure the telnet service does not run on this system?
A: YES

Q: Should Bastille ensure inetd's FTP services does not run on this sytem?
A: Yes

Q: Would you like to display "Authorized Use" messages at log-in time?
A: Yes

Q: Who is responsible for granting authorization to use this machine?
A: Use the name of one of the authorized administrator accounts. Preferably, if the competition scenario gives you the name of a company, use the company name.

Q: Would you like to put limits on system resource usage?
A: No

Q: Should we restrict console access to a small group of user accounts?
A: Yes (root)

Q: Would you like to add additional loggin?
A: Yes

Q: Do you have a remote logging host?
A: No

Q: Would you like to set up process accounting?
A: No

Q: Would you like to disable acpid and/or apmd?
A: Yes

Q: Would you like to deactivate NFS and Samba?
A: Yes (unless your competition scenario says otherwise!)

Q: Do you want to stop sendmail from running in daemon mode?
A: Yes

Q: Would you like to deactivate the Apache web server?
A: Yes (unless your competition scenario says otherwise!)

Q: Would you like to disable printing?
A: Yes (unless your competition scenario says otherwise!)

Q: Would you like to install TMPDIR/TMP scripts?
A: Yes

Q: Would you like to run the packet filtering script?
A: No

If for any reason you make a mistake, or you lose points, or Bastille locks you out of something, you can reverse Bastille's changes with:
sudo RevertBastille

You should also consider generating a vulnerability analysis report with Bastille using the following command:
sudo bastille --assessnobrowser

Chapter 5. S&D

Search & Destroy

Check for time-based scheduled events

****YOU CAN AUTOMATE THIS PART FROM CHAPTER %00**

First, we will check for cron jobs running under your user account.

crontab –l

```
cyberpatriot@ubuntu:~$ crontab -l
no crontab for cyberpatriot
```

Second, check for cron jobs running under root. Just prepend *sudo*.
sudo crontab –l

```
argus@cp-ub10-01:~$ sudo crontab -l
[sudo] password for argus:
*/5 * * * * export DISPLAY=:0 && /usr/bin/gnome-terminal -x /bin/bash -c "echo '
You have been hacked..'; sleep 5"
```

In the event that you find a scheduled task that you wish to remove from crontab, use the "-e" switch instead of "-l".
sudo crontab –e

Delete the scheduled jobs as necessary.

Check for remaining crontabs. These are located in the */var/spool/cron* directory.

sudo ls /var/spool/cron/crontabs

```
argus@cp-ub10-01:~$ sudo ls /var/spool/cron/crontabs
root
```

This means that there are crontabs assigned to the user *root*. View the contents of that file.

```
argus@cp-ub10-01:~$ sudo cat /var/spool/cron/crontabs/root
# DO NOT EDIT THIS FILE - edit the master and reinstall.
# (- installed on Wed Jul  3 14:58:34 2013)
# (Cron version -- $Id: crontab.c,v 2.13 1994/01/17 03:20:37 vixie Exp $)
*/5 * * * * export DISPLAY=:0 && /usr/bin/gnome-terminal -x /bin/bash -c "echo '
You have been hacked..'; sleep 5"
```

In this particular example, the file should be removed.

sudo rm /var/spool/cron/crontabs/root

Finally, check for any miscellaneous cron jobs in the */etc* directory.

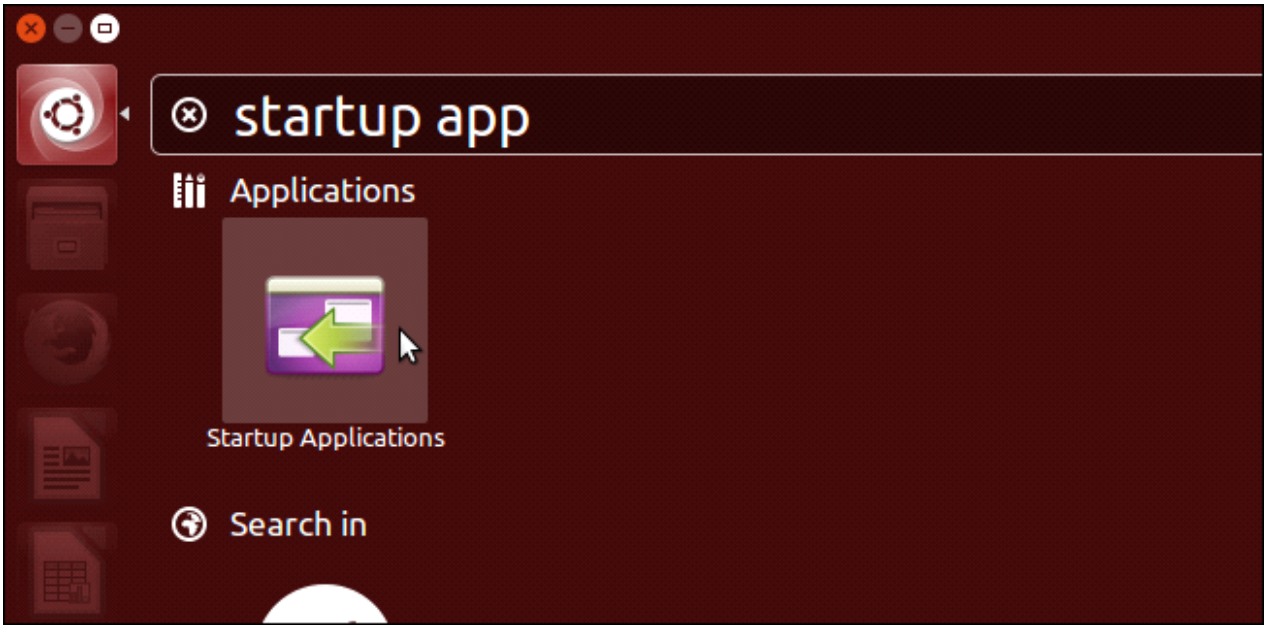
sudo ls –l /etc/cron*

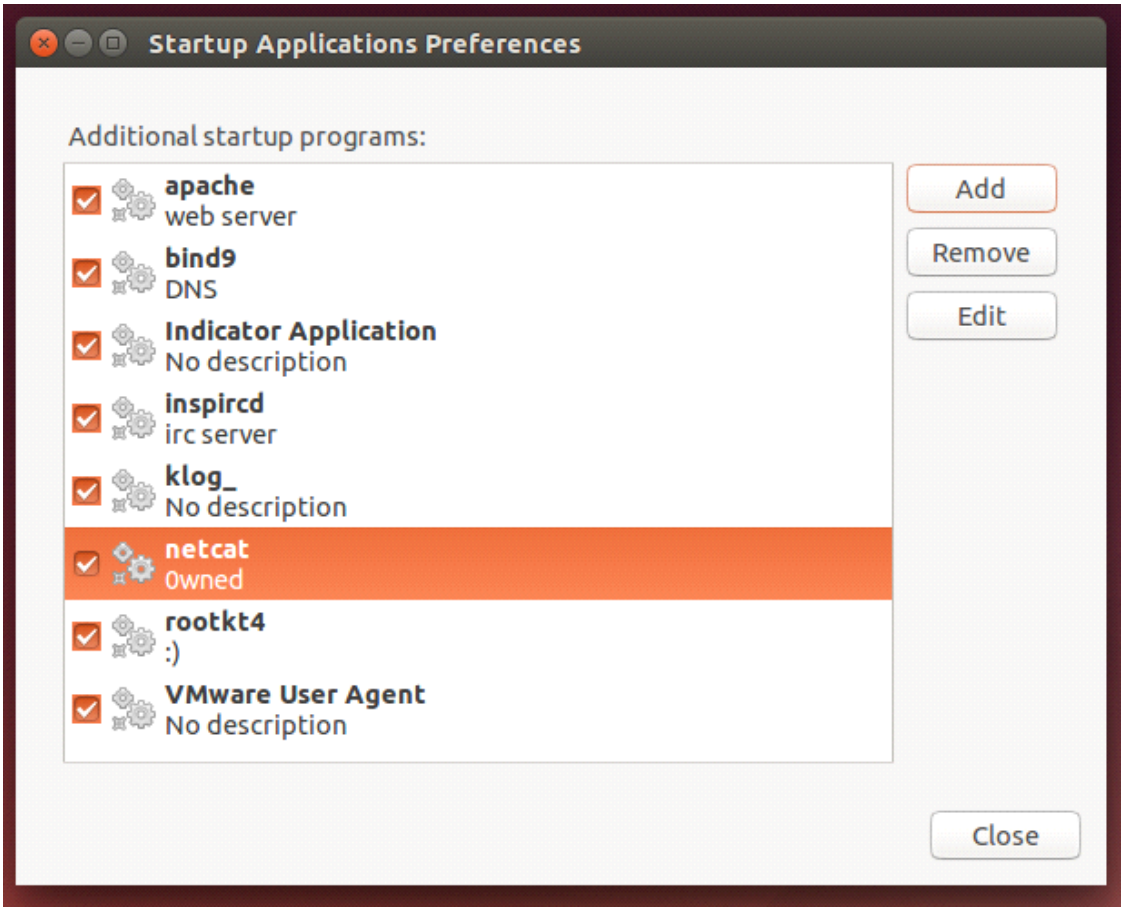
For each file listed, you should investigate its contents. If the task appears to be malicious, rm it.

Check startup directories for malicious files

sudo nano /etc/rc.local

```
cd /etc/init.d
ls -l
```





Finding Files with Unwanted Permissions

This section provides information on the use of the find command to find and display files and directories with permissions that may be undesirable. All examples in this section will output results

in the format typically provided by ls -l.

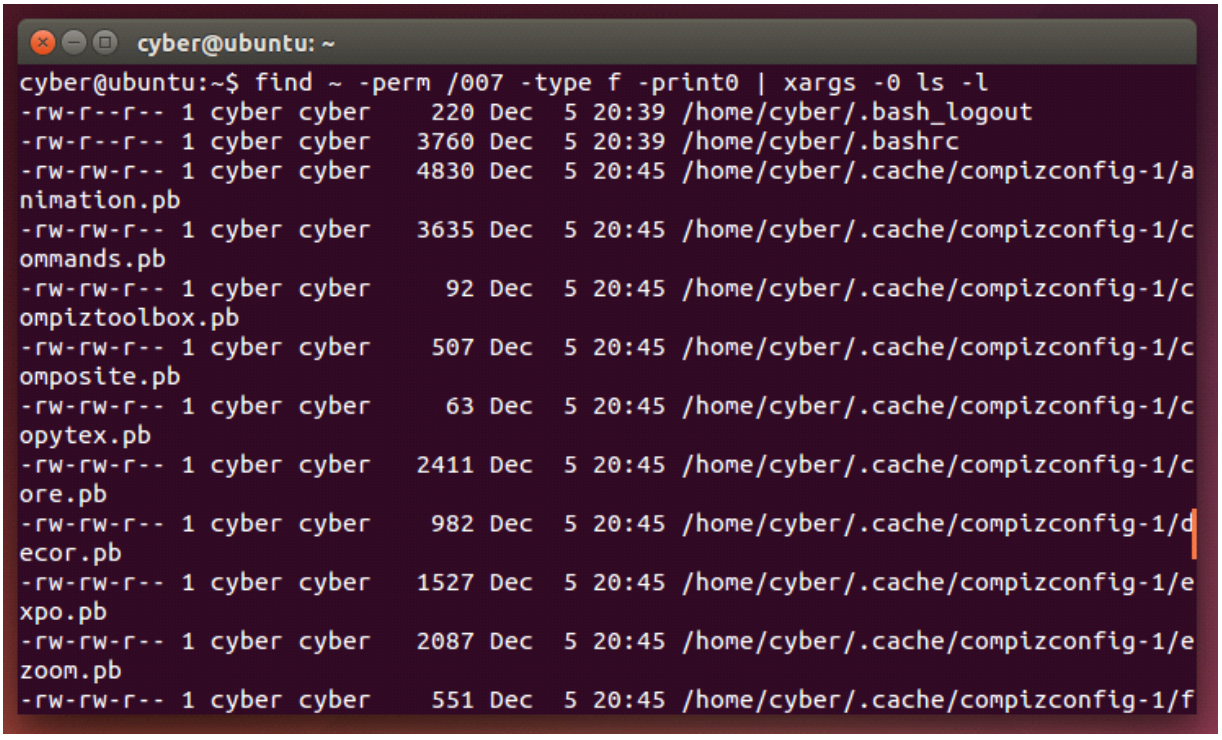
All examples will use the following format:

find /dir/name -perm /005 -type f -print0 | xargs -0 ls -l

Explanation of these options:
/dir/name — Directory being searched
-perm — Instructs **find** to match files based on given permissions
-type f — Instructs **find** to match only regular files
-print0 — Sends output to stdout using the null character
xargs — Command that builds another command to execute based on the content of stdin
-0 — Use the null character as the item delimiter

Find All Files Accessible by 'Other' in Any Way

find ~ -perm /007 -type f -print0 | xargs -0 ls -l



This will display all files within your home directory that have read, write, or execute permissions for 'other.' This is useful for getting a general idea of what others can access in your home directory.

Find All Files Writable by 'Other'

find /extra/research0 -perm /002 -type f -print0 | xargs -0 ls -l

This will display all files in the /extra/research0 directory that are writable by 'other.' Any files that appear in this list should be investigated, and in most cases, permissions should be changed to something more restrictive.

Find All Files Owned by a Specific Group

find ~ -group NameOfGroup -type f -print0 | xargs -0 ls -l

This will display all files in your home directory that are owned by the specified group.

Find All Files Not Owned by a Specific Group

find . -not -group NameOfGroup -type f -print0 | xargs -0 ls -l

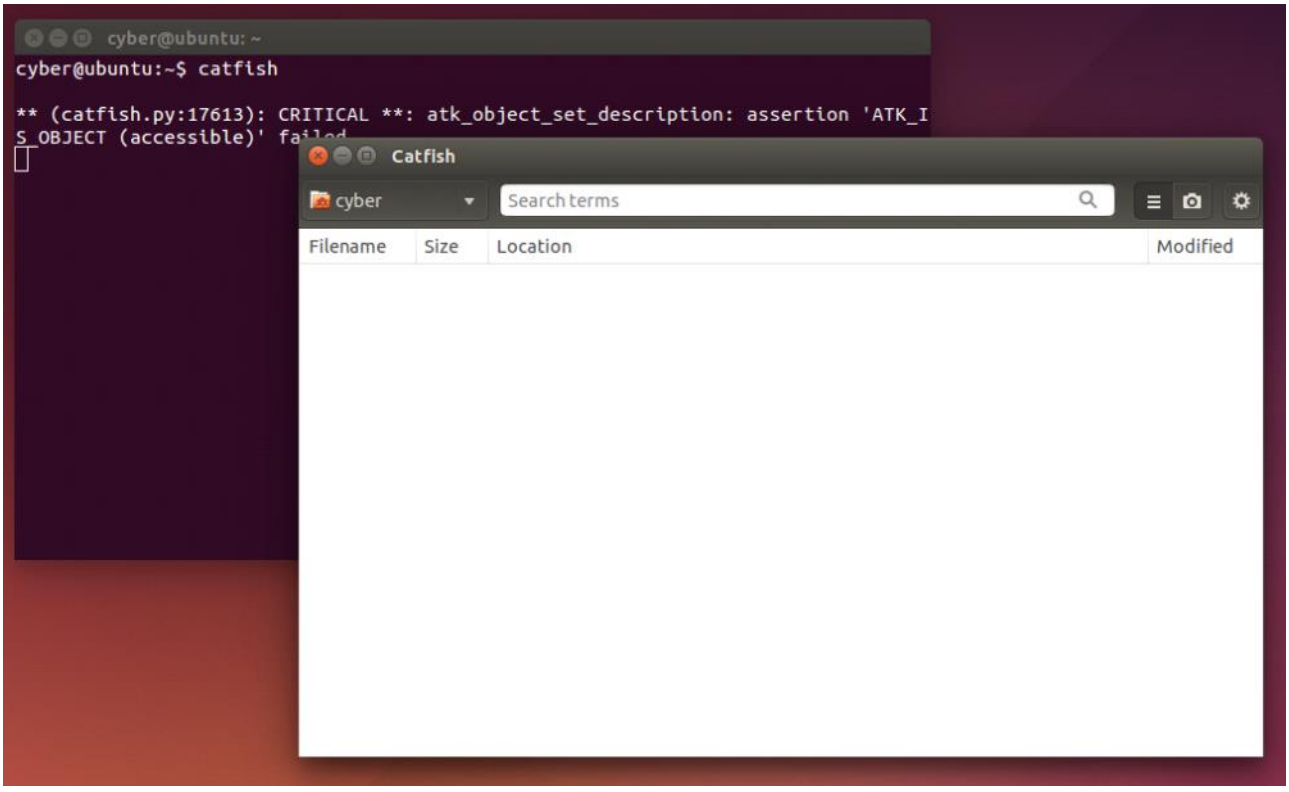
Finding file instructions from <http://www.ics.uci.edu/computing/linux/file-security.php>

Find and remove all unnecessary media files

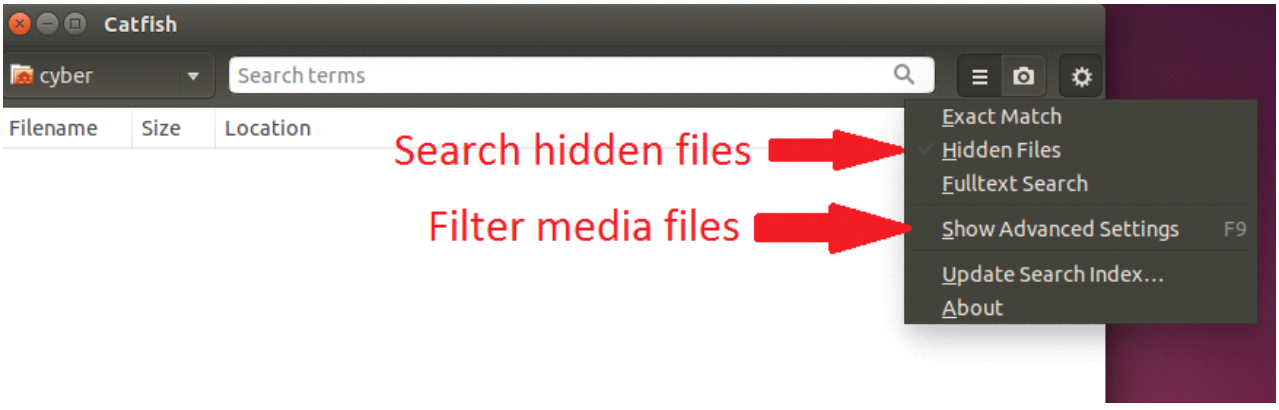
First, install *catfish*.

```
cyber@ubuntu: ~
cyber@ubuntu:~$ sudo apt-get install catfish
[sudo] password for cyber:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  python3-pexpect
Suggested packages:
  python-pexpect-doc
The following NEW packages will be installed:
  catfish python3-pexpect
0 upgraded, 2 newly installed, 0 to remove and 57 not upgraded.
Need to get 116 kB of archives.
After this operation, 1,020 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

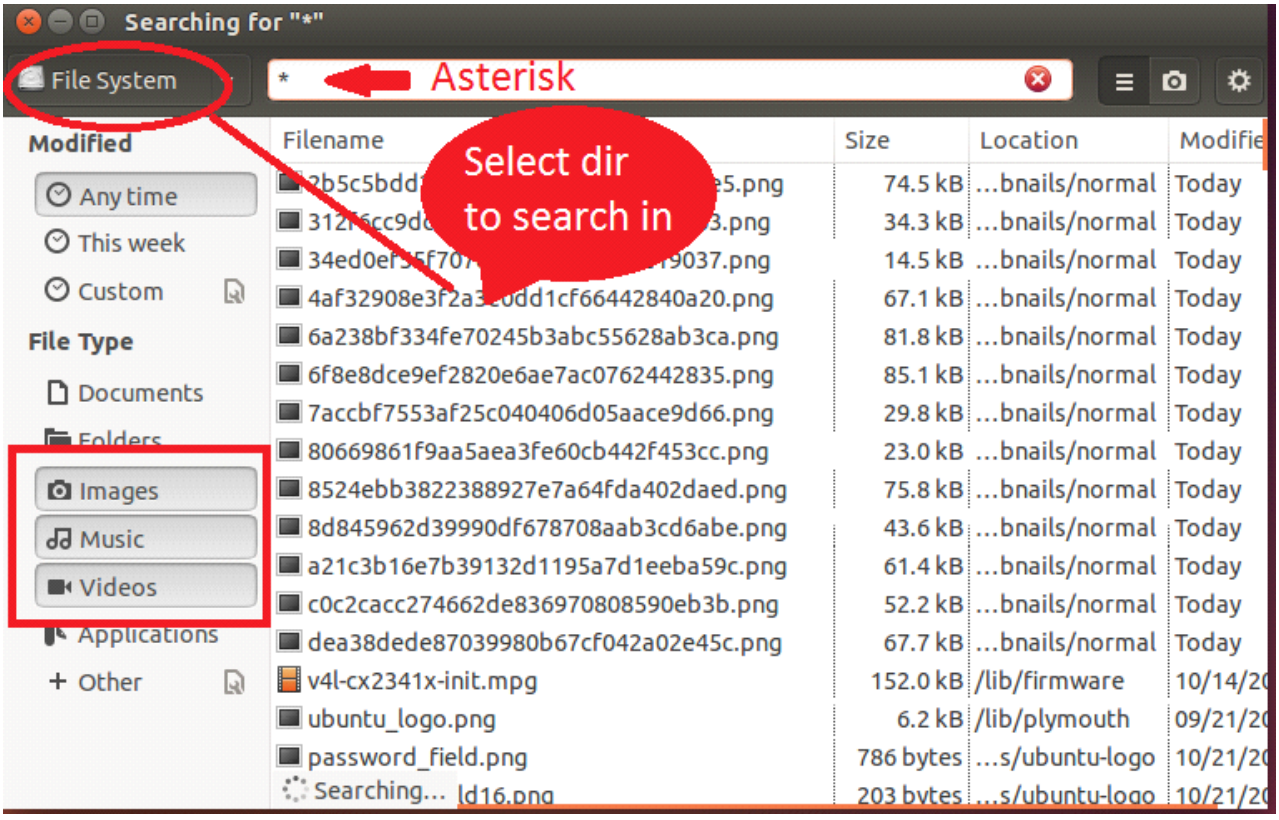
Next, run *catfish*.



Click the wrench in the upper right hand corner and select "Hidden Files" and "Show Advanced Settings."



Filter the *Images*, *Music*, and *Videos* filetypes and search for a wildcard. Select the filesystem directory in the upper lefthand corner.



Check and monitor logs

See **Log Paths** on pg. 10 of this guide.

Check logs easily and monitor them with either **logwatch** or **logcheck**.

sudo apt-get install logwatch

or

sudo apt-get install logcheck

```
          ~$ sudo apt-get install logwatch
[sudo] password for
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  libdate-manip-perl
Suggested packages:
  fortune-mod
The following NEW packages will be installed:
  libdate-manip-perl logwatch
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 1,381 kB of archives.
After this operation, 13.0 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://mirrors.linode.com/ubuntu/ trusty/main libdate-manip-perl all 6.42-1 [913 kB]
Get:2 http://mirrors.linode.com/ubuntu/ trusty/main logwatch all 7.4.0+svn20130529rev144-1ubuntu1 [468 kB]
Fetched 1,381 kB in 0s (13.7 MB/s)
Selecting previously unselected package libdate-manip-perl.
(Reading database ... 27302 files and directories currently installed.)
Preparing to unpack .../libdate-manip-perl_6.42-1_all.deb ...
Unpacking libdate-manip-perl (6.42-1) ...
Selecting previously unselected package logwatch.
Preparing to unpack .../logwatch_7.4.0+svn20130529rev144-1ubuntu1_all.deb ...
Unpacking logwatch (7.4.0+svn20130529rev144-1ubuntu1) ...
Processing triggers for man-db (2.6.7.1-1ubuntu1) ...
Setting up libdate-manip-perl (6.42-1) ...
Setting up logwatch (7.4.0+svn20130529rev144-1ubuntu1) ...
          ~$ logwatch
File /var/log/mail.log is not readable.  Check permissions.  You are not running as superuser.
File /var/log/mail.log.1 is not readable.  Check permissions.  You are not running as superuser.
gzip: /var/log/syslog.2.gz: Permission denied
system 'zcat '/var/log/syslog.2.gz' >> /tmp/logwatch.vMblpBhh/syslog-archive' failed: 256 at /usr/sbin/logwatch line 774.
          :~$ sudo logwatch
```

```
--------------------- SSHD Begin ------------------------


Disconnecting after too many authentication failures for user:
   root : 130 Time(s)

Failed logins from:
   74.79.237.112 (cpe-74-79-237-112.twcny.res.rr.com): 1 time
   95.191.220.93 (b-internet.95.191.220.93.nsk.sibirtelecom.ru): 1 time
   103.41.124.15: 1162 times
   103.41.124.34: 1593 times
   103.41.124.48: 1570 times
   103.232.35.19: 2 times
   111.161.126.234 (dns234.online.tj.cn): 75 times
   122.225.97.66: 30 times
   122.225.97.75: 30 times
   122.225.97.81 (huzhou.ctc.mx.fund123.cn): 23 times
   122.225.97.82 (huzhou.ctc.mx.fund123.cn): 36 times
   122.225.97.89 (huzhou.ctc.mx.fund123.cn): 9 times
   122.225.109.112: 12 times
   188.255.111.126 (broadband-188-255-111-126.nationalcablenetworks.ru): 1 time
   213.165.85.185 (s15313466.onlinehome-server.info): 1 time
   222.243.209.150: 1 time

Illegal users from:
   undef: 48 times
   5.228.57.226 (broadband-5-228-57-226.nationalcablenetworks.ru): 1 time
   12.152.123.160: 2 times
   37.204.60.245 (broadband-37.204-60-245.nationalcablenetworks.ru): 1 time
   40.130.45.18 (h18.45.130.40.static.ip.windstream.net): 1 time
   46.242.20.152 (broadband-46-242-20-152.nationalcablenetworks.ru): 1 time
   46.242.68.135 (broadband-46-242-68-135.nationalcablenetworks.ru): 1 time
   62.84.99.38 (pppoe-62-84-99-38.dubna.ru): 1 time
   63.235.180.46 (63-235-180-46.dia.static.qwest.net): 1 time
   64.127.92.166 (cap31-ccu1-sims-92-166.hamiltoncom.net): 1 time
   64.130.128.63 (64-130-128-63.pool.dsl.scrtc.com): 1 time
   64.130.145.127 (64-130-145-127.pool.dsl.scrtc.com): 1 time
   67.82.118.245 (ool-435276f5.dyn.optonline.net): 1 time
   72.47.120.243 (doc-72-47-120-243.kellogg.id.cebridge.net): 1 time
   74.68.38.200 (cpe-74-68-38-200.nj.res.rr.com): 1 time
   91.143.29.35: 1 time
   98.235.163.182 (c-98-235-163-182.hsd1.pa.comcast.net): 1 time
   103.232.35.19: 8 times
   108.48.173.3 (static-108-48-173-3.washdc.fios.verizon.net): 1 time
   109.184.56.113 (109-184-56-113.dynamic.mts-nn.ru): 2 times
   111.161.126.234 (dns234.online.tj.cn): 39 times
   166.154.164.109 (109.sub-166-154-164.myvzw.com): 1 time
   174.106.49.197 (cpe-174-106-049-197.ec.res.rr.com): 1 time
   176.51.106.221 (b-internet.176.51.106.221.nsk.rt.ru): 1 time
   176.106.9.212: 1 time
   204.10.76.214 (ds.activeserve.com): 1 time
   207.10.147.6: 1 time
   212.164.129.74: 1 time
   213.165.85.185 (s15313466.onlinehome-server.info): 10 times


Received disconnect:
   11:  [preauth] : 4324 Time(s)
   11: Bye Bye [preauth] : 125 Time(s)

**Unmatched Entries**
message repeated 2 times: [ Failed password for root from 103.41.124.34 port 38873 ssh2] : 1 time(s)
message repeated 2 times: [ Failed password for root from 103.41.124.48 port 59936 ssh2] : 1 time(s)
message repeated 2 times: [ Failed password for root from 103.41.124.34 port 36218 ssh2] : 1 time(s)
message repeated 2 times: [ Failed password for root from 103.41.124.15 port 32772 ssh2] : 1 time(s)
message repeated 2 times: [ Failed password for root from 103.41.124.34 port 40656 ssh2] : 1 time(s)
message repeated 2 times: [ Failed password for root from 103.41.124.34 port 56502 ssh2] : 1 time(s)
```

NOTE: I would have probably censored the IP addresses if they were not trying to break into my box.

⭐ Install ClamAV or Bitdefender or lmd

****YOU CAN AUTOMATE THIS PART FROM CHAPTER %00**

To install ClamAV, run **apt-get install clamav**

To install Bitdefender, download and add the BitDefender pgp key so apt can verify the

BitDefender pkg

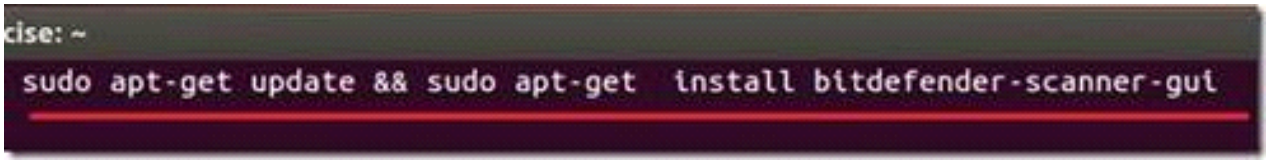
```
wget -O- -q http://download.bitdefender.com/repos/deb/bd.key.asc | sudo apt-key add -
```

Second, add BitDefender's repo to your *sources.list* file

```
sudo sh -c 'echo "deb http://download.bitdefender.com/repos/deb/ bitdefender non-free" >>
/etc/apt/sources.list'
```

Then, update apt and install bitdefender

```
sudo apt-get update && sudo apt-get  install bitdefender-scanner-gui
```



Navigate to tty7 and launch the BitDefender GUI. Run an update and then scan /

Source: <http://www.liberiangeek.net/2012/07/install-bitdefender-from-its-software-repository-in-ubuntu-12-04/>

To install lmd (Linux Malware Detector, which is preferable):

wget *http://rfxn.com/downloads/maldetect-current.tar.gz*

Then you will need to will need to extract the tar zip

tar zxvf maldetect*

Run the install script and then use

lmd --help

For more information.

Install chkrootkit

****YOU CAN AUTOMATE THIS PART FROM CHAPTER %00**

sudo apt-get install chkrootkit

```
w@ubuntu:~$ sudo apt-get install chkrootkit
```

Then, initiate a chkrootkit scan. Output the contents to a log file.

```
w@ubuntu:~$ sudo chkrootkit > log
```

Examine the contents of the "log" file and investigate anything malicious. Be mindful of false positives!

Install rkhunter

****YOU CAN AUTOMATE THIS PART FROM CHAPTER %00**

sudo apt-get install rkhunter

```
w@ubuntu:~$ sudo apt-get install rkhunter
```

Then, make sure rkhunter is up to date with the *--update* option.

sudo rkhunter --update

```
w@ubuntu:~$ sudo rkhunter --update
[ Rootkit Hunter version 1.3.8 ]

Checking rkhunter data files...
  Checking file mirrors.dat                                  [ No update ]
  Checking file programs_bad.dat                             [ Updated ]
  Checking file backdoorports.dat                            [ No update ]
  Checking file suspscan.dat                                 [ No update ]
```

Finally, initiate the rkhunter scan.

sudo rkhunter --check

```
w@ubuntu:~$ sudo rkhunter --check
[ Rootkit Hunter version 1.3.8 ]

Checking system commands...

  Performing 'strings' command checks
```

Tip: Check on the scan periodically, as you will be required to hit [Enter] to continue to the next phase of the scan.

Once the scan finishes, navigate to */var/log/* and investigate the *rkhunter.log* file. Respond to anything malicious and be mindful of false positives.

```
System checks summary
=====================

File properties checks...
    Files checked: 130
    Suspect files: 1

Rootkit checks...
    Rootkits checked : 245
    Possible rootkits: 0

Applications checks...
    All checks skipped

The system checks took: 8 minutes and 11 seconds

All results have been written to the log file (/var/log/rkhunter.log)
```

Download psad

First, we need to create a directory to store the contents of psad in.

mkdir psad

From there, **cd** into that directory and **wget** the tar.gz

wget http://cipherdyne.org/psad/download/**psad-2.2.tar.gz**

```
w@ubuntu:~$ mkdir psad
mkdir: cannot create directory `psad': File exists
w@ubuntu:~$ cd psad
w@ubuntu:~/psad$ wget http://cipherdyne.org/psad/download/psad-2.2.tar.gz
--2013-12-03 13:34:45--  http://cipherdyne.org/psad/download/psad-2.2.tar.gz
Resolving cipherdyne.org (cipherdyne.org)... 74.220.215.85
Connecting to cipherdyne.org (cipherdyne.org)|74.220.215.85|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1525092 (1.5M) [application/x-gzip]
Saving to: `psad-2.2.tar.gz'

100%[=====================================>] 1,525,092   3.31M/s   in 0.4s

2013-12-03 13:34:47 (3.31 MB/s) - `psad-2.2.tar.gz' saved [1525092/1525092]
```

Extract the archive using **tar**

```
w@ubuntu:~/psad$ tar zxvf psad-2.2.tar.gz
```

And then execute the install script as root.

```
w@ubuntu:~/psad$ cd psad-2.2/
w@ubuntu:~/psad/psad-2.2$ sudo ./install.pl
```

The installation will ask a few questions. Answer as so:

```
[+] psad alerts will be sent to:

       root@localhost

[+] Would you like alerts sent to a different address ([y]/n)?  n

[+] By default, psad parses all iptables log messages for scan activity.
    However, psad can be configured to only parse those iptables messages
    that match particular strings (that are specified in your iptables
    ruleset with the --log-prefix option).

    Would you like psad to only parse specific strings in iptables
    messages (y/[n])?  n
```

```
[+] By default, psad matches Snort rules against any IP addresses, but
    psad offers the ability to restrict signature matches to specific
    networks with the HOME_NET variable (similar to Snort).  However, psad
    also offers the ability to acquire all local subnets on the local system
    by parsing the output of "ifconfig", or the subnets can be restricted
    to a limited set of networks.

    First, is it ok to leave the HOME_NET setting as "any" ([y]/n)?  y

[+] psad has the capability of sending scan data via email alerts to the
    DShield distributed intrusion detection system (www.dshield.org).  By
    default this feature is not enabled since firewall log data is sensitive,
    but submitting logs to DShield provides a valuable service and assists
    in generally enhancing internet security.  As an optional step, if you
    have a DShield user id you can edit the "DSHIELD_USER_ID" variable
    in /etc/psad/psad.conf

    Would you like to enable DShield alerts (y/[n])?  n
```

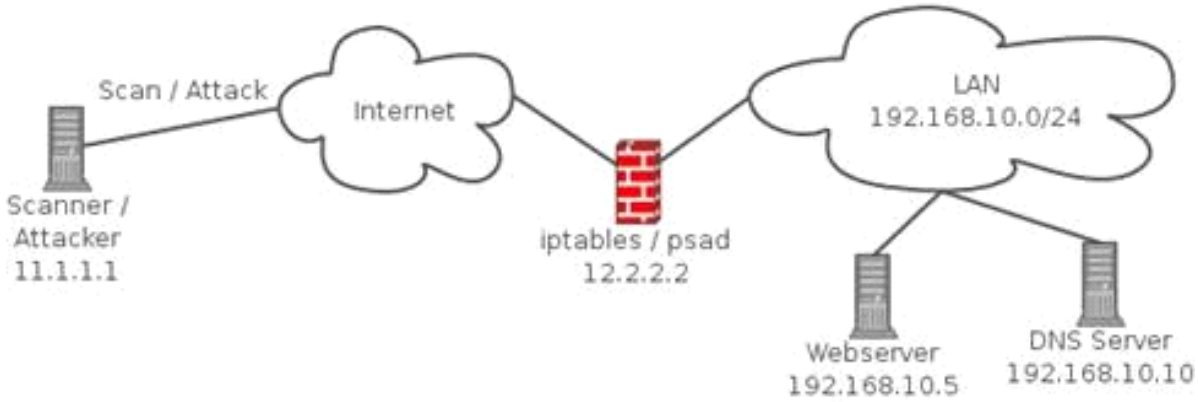
```
    Would you like to install the latest signatures from
      http://www.cipherdyne.org/psad/signatures (y/n)?  y
[+] Downloading latest signatures from:
        http://www.cipherdyne.org/psad/signatures
--2013-12-03 13:31:06--  http://www.cipherdyne.org/psad/signatures
Resolving www.cipherdyne.org (www.cipherdyne.org)... 74.220.215.85
Connecting to www.cipherdyne.org (www.cipherdyne.org)|74.220.215.85|:80... conn
ected.
HTTP request sent, awaiting response... 200 OK
Length: 45096 (44K) [text/plain]
Saving to: `signatures'

100%[===================================>] 45,096      --.-K/s   in 0.001s

2013-12-03 13:31:10 (29.7 MB/s) - `signatures' saved [45096/45096]
```

Finally, start psad with the **service** command.

```
w@ubuntu:~/psad/psad-2.2$ sudo service psad start
Starting psad: [*] Could not find mail, edit /etc/psad/psad.conf at /usr/sbin/p
sad line 10382.
```



Install tiger

sudo apt-get install tiger

```
w@ubuntu:~$ sudo apt-get install tiger
```

Then, initiate a tiger scan: sudo tiger

```
w@ubuntu:~$ sudo tiger
Tiger UN*X security checking system
```

Once complete, traverse to the */var/log/tiger* directory and inspect the report. If you **cd** to this directory, you might notice a *Permission Denied* error.

```
w@ubuntu:~$ cd /var/log/tiger
bash: cd: /var/log/tiger: Permission denied
```

This is due to the permissions of the directory for purposes of additional security. Use privilege escalation to view the necessary files.

```
w@ubuntu:~$ cd /var/log/tiger
bash: cd: /var/log/tiger: Permission denied
w@ubuntu:~$ sudo ls /var/log/tiger
security.report.ubuntu.tmp.10306
w@ubuntu:~$ sudo cat /var/log/tiger/security.report.ubuntu.tmp.10306|less
```

Cross reference the tiger codes found in the log file with **tigexp** for more information about the problem and how to fix it.

```
w@ubuntu:/var/log$ sudo cat tiger/security.report.ubuntu.tmp.10306|less
w@ubuntu:/var/log$ sudo tigexp pass012w

The listed home directory is specified for multiple users. This can
lead to denial-of-service and unexpected resource usage (i.e. shell
initialization files, etc) if not corrected.
```

Install lynis

****YOU CAN AUTOMATE THIS PART FROM CHAPTER %00**

First, we need to create a directory to store the contents of lynis in.

mkdir lynis

From there, **cd** into that directory and **wget** the tar.gz

wget **http://cisofy.com/files/lynis-1.3.6.tar.gz**

```
w@ubuntu:~$ mkdir lynis
w@ubuntu:~$ cd lynis
w@ubuntu:~/lynis$ wget http://cisofy.com/files/lynis-1.3.6.tar.gz
--2013-12-03 13:13:29--  http://cisofy.com/files/lynis-1.3.6.tar.gz
Resolving cisofy.com (cisofy.com)... 149.210.134.182
Connecting to cisofy.com (cisofy.com)|149.210.134.182|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 134121 (131K) [application/octet-stream]
Saving to: `lynis-1.3.6.tar.gz'

100%[=====================================>] 134,121      --.-K/s   in 0.004s

2013-12-03 13:13:31 (30.3 MB/s) - `lynis-1.3.6.tar.gz' saved [134121/134121]
```

Extract the archive using **tar**

```
w@ubuntu:~/lynis$ tar zxvf lynis-1.3.6.tar.gz
```

And then run the shell script as root, with the **-c** switch.

```
w@ubuntu:~/lynis$ cd lynis-1.3.6/
w@ubuntu:~/lynis/lynis-1.3.6$ sudo sh lynis -c

[ Lynis 1.3.6 ]
```

Finally, review the **lynis.log** file in */var/log*

Download and run fsaudit (file_system audit)

First, download the perl script.

wget **http://www.cise.ufl.edu/~jfh/jst/fsaudit**

Then launch the script as an administrator.

sudo perl fsaudit

View system processes and resource usage

sudo top

```
top - 13:51:00 up  1:43,  4 users,  load average: 3.69, 3.89, 3.98
Tasks: 211 total,   2 running, 209 sleeping,   0 stopped,   0 zombie
Cpu(s):  4.3%us, 93.8%sy,  0.5%ni,  0.8%id,  0.3%wa,  0.0%hi,  0.2%si,  0.0%st
Mem:   3066252k total,  2541876k used,   524376k free,   117324k buffers
Swap:  3105788k total,        0k used,  3105788k free,  1903868k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 1312 root      20   0  104m  38m 7012 S    8  1.3   3:38.23 Xorg
10979 root      20   0  2240  428  268 S    5  0.0   2:07.17 find_files
10976 root      26   6  2240  620  524 S    4  0.0   2:15.69 flogit
 2832 w         20   0 92008  16m  11m S    1  0.5   0:22.41 gnome-terminal
10731 root      20   0  2372  492  236 S    1  0.0   0:00.04 deb_checkmd5sum
 2498 w         20   0  343m 124m  30m S    1  4.2   0:54.64 compiz
```

View processes listening on ports

****YOU CAN AUTOMATE THIS PART FROM CHAPTER %00**

sudo netstat -plunt

```
w@ubuntu:~$ sudo netstat -tulnp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
 PID/Program name
tcp        0      0 127.0.0.1:587           0.0.0.0:*               LISTEN
 5105/sendmail: MTA:
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN
 17038/dnsmasq
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
 645/sshd
```

Tip: sshd is the program name, **645** is the process ID, **22** is the port it is listening on over **TCP**.

Secure ports

sudo ss -ln

If a port has 127.0.0.1:[port] in the line, it is not exposed. This can be ignored.
Else, there should only be ports that correspond to the critical services in your scenario!

When you find a port that you want closed, for example, 900:

sudo lsof -I :900

Find where the program is that was listening on the port:

sudo whereis nc

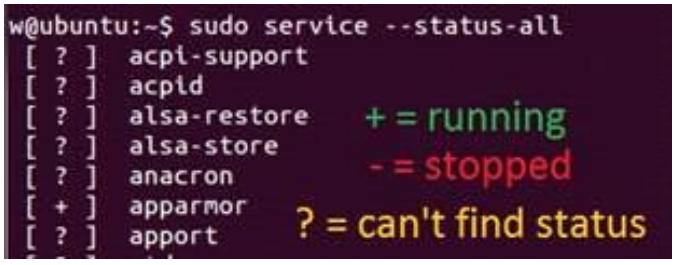
Delete with **rm** and **sudo apt-get purge**.

Finally, check to make sure that the port is closed:

sudo ss -l

★ View running services

sudo service --status-all

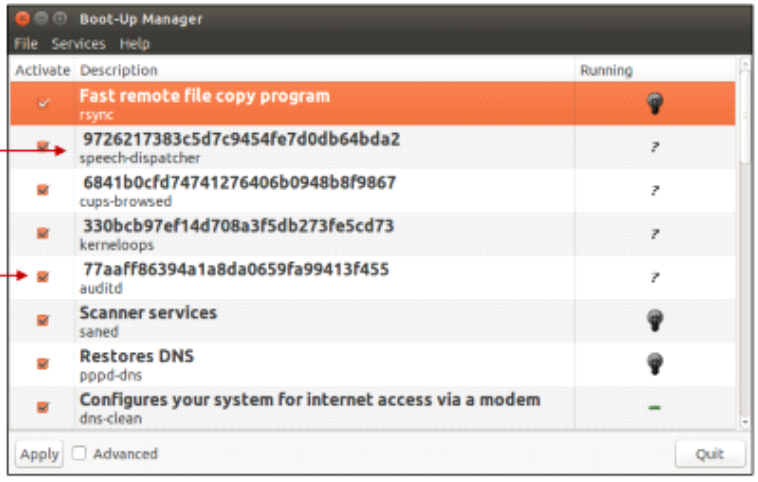


Services

- Can be viewed and managed in the GUI
- To install, type `apt-get install bum in` Terminal
- After installing, type `bum` to run

To start a service, right-click it and select "Start"

To enable a service, check the box next to it



When a service is started, the light bulb will light up. When stopped, the light bulb will be dark.

© Air Force Association

To lookup the process ID of a program name

sudo pgrep <programName>

```
w@ubuntu:~$ sudo netstat -tulnp|grep nc
tcp        0      0 0.0.0.0:8080            0.0.0.0:*               LISTEN
 30450/nc
w@ubuntu:~$ sudo pgrep nc
1248
2470
30450
```

To lookup the program name of a process ID

sudo ps -p <pid>

```
w@ubuntu:~$ pgrep nc
1248
2470
11943
w@ubuntu:~$ ps -p 11943
  PID TTY          TIME CMD
11943 pts/2    00:00:00 nc
```

To kill a process ID

sudo kill -s 9 <pid>

```
w@ubuntu:~$ nc -l 8080
```

```
w@ubuntu:~$ sudo pgrep nc
1248
2470
30450
w@ubuntu:~$ sudo kill -s 9 30450
```

```
w@ubuntu:~$ nc -l 8080
Killed
```

To kill a program by name

sudo kill -s 9 <programName>

```
w@ubuntu:~$ nc -l 7071
```

```
w@ubuntu:~$ sudo netstat -tulnp|grep nc
tcp        0      0 0.0.0.0:7071            0.0.0.0:*               LISTEN
 11927/nc
w@ubuntu:~$ sudo killall -s 9 nc
```

```
w@ubuntu:~$ nc -l 7071
Killed
```

Download checksec

wget http://www.trapkit.de/tools/checksec.sh

Then, make the script executable and initiate it with the *--proc-all option*.

```
w@ubuntu:~$ wget http://www.trapkit.de/tools/checksec.sh
--2013-12-03 14:10:58--  http://www.trapkit.de/tools/checksec.sh
Resolving www.trapkit.de (www.trapkit.de)... 81.169.145.67, 2a01:238:20a:202:10
86::86
Connecting to www.trapkit.de (www.trapkit.de)|81.169.145.67|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 27095 (26K) [application/x-sh]
Saving to: `checksec.sh'

100%[=====================================>] 27,095      --.-K/s   in 0s

2013-12-03 14:11:00 (86.6 MB/s) - `checksec.sh' saved [27095/27095]

w@ubuntu:~$ sudo chmod +x checksec.sh
w@ubuntu:~$ sudo ./checksec.sh --proc-all
* System-wide ASLR (kernel.randomize_va_space): On (Setting: 2)
```

Uninstall malicious and unneeded software using **apt-get remove <package>**.

```
w@ubuntu:~$ sudo apt-get remove telnet
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  thunderbird-globalmenu
Use 'apt-get autoremove' to remove them.
The following packages will be REMOVED:
  telnet
0 upgraded, 0 newly installed, 1 to remove and 148 not upgraded.
After this operation, 193 kB disk space will be freed.
Do you want to continue [Y/n]? y
```

Tip: Uninstall multiple packages at once by adding spaces. Example: sudo apt-get remove telnet nginx mysql

Port Scanning and Vulnerability Scanning

⭐ Check open ports with netstat

****YOU CAN AUTOMATE THIS PART FROM CHAPTER %00**

netstat -plunt

⭐ Scanning with nmap

Install nmap:

sudo apt-get install nmap

Scan for every open TCP and UDP port:

sudo nmap -n -PN -sT -sU -p- 127.0.0.1

⭐ Scanning with OpenVAS

Link to instructions: https://hackertarget.com/install-openvas-7-ubuntu/

Chapter 6.
INTRUSION DETECTION AND NETWORK MONITORING

Install a HIDS

Install tripwire

By Justin Ellingwood

Introduction

Security is an incredibly complex problem when administering online servers. While it is possible to configure firewalls, fail2ban policies, secure services, and lock down applications, it is difficult to know for sure if you have effectively blocked every attack.

A host-based intrusion detection system (HIDS), works by collecting details about your computer's filesystem and configuration. It then stores this information to reference and validate the current state of the system. If changes are found between the known-good state and the current state, it could be a sign that your security has been compromised.

A popular host-based intrusion detection system on Linux is **tripwire**. This software can keep track of many different filesystem data points in order to detect whether unauthorized changes have occurred.

Install Tripwire

Fortunately, tripwire can be found in Ubuntu's default repositories. We can install it with apt-get by typing:
sudo apt-get update
sudo apt-get install tripwire
This installation will run through quite a bit of configuration of the packages that are required.

First, it will configure the mail application that is being pulled in as a dependency. If you want to configure email notifications, select "internet site".

It will ask you if you want to select passphrases during installation. Select "yes" to both of these prompts. It will ask if it can rebuild the configuration file. Select "yes".

It will ask a similar question about the policy file. Again, answer "yes".

Next, you will be asked to choose and confirm a site key passphrase. Tripwire uses two keys to secure its configuration files.

- **site key**: This key is used to secure the configuration files. We need to ensure that the configuration files aren't modified, or else our entire detection system cannot be trusted. Since the same configuration files can be used for multiple servers, this key can be used across servers.
- **local key**: This key is used on each machine to run the binaries. This is necessary to ensure that our binaries are not run without our consent.

You will first choose and confirm a passphrase for the site key, and then for the local key. Make sure you choose strong passphrases.

Initialize the Database

Following the installation, you must initialize and configure your installation. Like most security programs, tripwire is shipped with generic, but strict defaults that may need to be fine-tuned for your specific installation.

First, if you did not choose yes to create a policy file during installation, you can do so now by issuing the command:
sudo twadmin --create-polfile /etc/tripwire/twpol.txt

You will be prompted for the site passphrase you configured earlier.

This creates an encrypted policy file from the plain text one that we specified in the /etc/tripwire/ directory. This encrypted file is what tripwire actually reads when running its checks.

We can now initialize the database that tripwire will use to validate our system. This uses the policy file that we just initiated and checks the points that are specified within.

Because this file has not been tailored for our system yet, we will have a lot of warnings, false positives, and errors. We will use these as a reference to fine-tune our configuration file in a moment.

The basic way to initialize the database is by running:
sudo tripwire --init

This will create our database file and complain about the things that we must adjust in the configuration.

Because we want to save the results to inform our configuration decisions, we can grab any instance where it mentions a file, and put it in a file in the tripwire configuration directory. We can run the check and place the files listed into a file called test_results in our tripwire config directory:
sudo sh -c 'tripwire --check | grep Filename > test_results'

If we view this file, we should see entries that look like this:

```
less /etc/tripwire/test_results
```

```
Filename: /etc/rc.boot
Filename: /root/mail
Filename: /root/Mail
Filename: /root/.xsession-errors
. . .
```

Configure the Policy File to Match Your System

Now that we have a list of files that are setting off tripwire, we can go through our policy file and edit it to get rid of these false positives.
Open the plain text policy in your editor with root privileges:
sudo nano /etc/tripwire/twpol.txt
Do a search for each of the files that were returned in the test_results file. Comment out all of the lines that you find that match.
In the "Boot Scripts" section, you should comment out the /etc/rc.boot line, since this isn't present in an Ubuntu system:

```
(
  rulename = "Boot Scripts",
  severity = $(SIG_HI)
)
{
        /etc/init.d             -> $(SEC_BIN) ;
        #/etc/rc.boot            -> $(SEC_BIN) ;
        /etc/rcS.d              -> $(SEC_BIN) ;
```

There were a lot of files in the /root home directory that needed to be commented out on my system.
Anything that is not present on your system should be commented out:

```
(
  rulename = "Root config files",
  severity = 100
)
{
        /root                           -> $(SEC_CRIT) ; # Catch all additions to /root
        #/root/mail                     -> $(SEC_CONFIG) ;
        #/root/Mail                     -> $(SEC_CONFIG) ;
        #/root/.xsession-errors         -> $(SEC_CONFIG) ;
        #/root/.xauth                   -> $(SEC_CONFIG) ;
        #/root/.tcshrc                  -> $(SEC_CONFIG) ;
        #/root/.sawfish                 -> $(SEC_CONFIG) ;
        #/root/.pinerc                  -> $(SEC_CONFIG) ;
        #/root/.mc                      -> $(SEC_CONFIG) ;
        #/root/.gnome_private           -> $(SEC_CONFIG) ;
        #/root/.gnome-desktop           -> $(SEC_CONFIG) ;
        #/root/.gnome                   -> $(SEC_CONFIG) ;
        #/root/.esd_auth                        -> $(SEC_CONFIG) ;
        #/root/.elm                     -> $(SEC_CONFIG) ;
        #/root/.cshrc                   -> $(SEC_CONFIG) ;
        /root/.bashrc                   -> $(SEC_CONFIG) ;
        #/root/.bash_profile            -> $(SEC_CONFIG) ;
        #/root/.bash_logout             -> $(SEC_CONFIG) ;
        /root/.bash_history             -> $(SEC_CONFIG) ;
        #/root/.amandahosts             -> $(SEC_CONFIG) ;
        #/root/.addressbook.lu          -> $(SEC_CONFIG) ;
        #/root/.addressbook             -> $(SEC_CONFIG) ;
        #/root/.Xresources              -> $(SEC_CONFIG) ;
        #/root/.Xauthority              -> $(SEC_CONFIG) -i ; # Changes Inode number on log
        #/root/.ICEauthority                -> $(SEC_CONFIG) ;
}
```

The last part of my check was complaining about file descriptors in the /proc filesystem. These files change all of the time, so will trigger false positives regularly if we leave the configuration as is.

In the "Devices & Kernel information" section, you can see that the /proc filesystem is listed to be checked.

```
(
  rulename = "Devices & Kernel information",
  severity = $(SIG_HI),
)
{
        /dev            -> $(Device) ;
        /proc           -> $(Device) ;
}
```

However, this will check every file under it. We don't particularly want that. Instead, we will remove this specification, and add configuration options for all of the directories under /proc that we *do* want to check:

```
{
        /dev                    -> $(Device) ;
        #/proc                  -> $(Device) ;
        /proc/devices           -> $(Device) ;
        /proc/net               -> $(Device) ;
        /proc/tty               -> $(Device) ;
        /proc/sys               -> $(Device) ;
        /proc/cpuinfo           -> $(Device) ;
        /proc/modules           -> $(Device) ;
        /proc/mounts            -> $(Device) ;
        /proc/dma               -> $(Device) ;
        /proc/filesystems       -> $(Device) ;
        /proc/interrupts        -> $(Device) ;
        /proc/ioports           -> $(Device) ;
        /proc/scsi              -> $(Device) ;
        /proc/kcore             -> $(Device) ;
        /proc/self              -> $(Device) ;
        /proc/kmsg              -> $(Device) ;
        /proc/stat              -> $(Device) ;
        /proc/loadavg           -> $(Device) ;
        /proc/uptime            -> $(Device) ;
        /proc/locks             -> $(Device) ;
        /proc/meminfo           -> $(Device) ;
        /proc/misc              -> $(Device) ;
}
```

While we are in this portion of the file, we also want to do something with the /dev/pts filesystem. Tripwire will not check that location by default because it is told to check /dev, and /dev/pts is on a separate filesystem, which it will not enter unless specified. To get tripwire to check this as well, we can explicitly name it here:

```
{
        /dev                    -> $(Device) ;
        /dev/pts                -> $(Device) ;
        #/proc                  -> $(Device) ;
        /proc/devices           -> $(Device) ;
        /proc/net               -> $(Device) ;
        /proc/tty               -> $(Device) ;
        . . .
```

The last thing we will comment out are the **/var/run** and **/var/lock** lines so that our system does not flag normal filesystem changes by services:

```
(
  rulename = "System boot changes",
  severity = $(SIG_HI)
)
{
        #/var/lock              -> $(SEC_CONFIG) ;
        #/var/run               -> $(SEC_CONFIG) ; # daemon PIDs
        /var/log                -> $(SEC_CONFIG) ;
}
```

Save and close the file when you are finished editing.

Now that our file is configured, we need to implement it by recreating the encrypted policy file that tripwire actually reads:
sudo twadmin -m P /etc/tripwire/twpol.txt
After this is created, we must reinitialize the database to implement our policy:
sudo tripwire --init
All of the warnings that you received earlier should be gone now. If there are still warnings, you should continue editing your /etc/tripwire/twpol.txt file until they are gone.

Verify the Configuration

If your database initialization didn't complain about any files, then your configuration should match your system at this point.

But we should run a check to see what the tripwire report looks like and if there are truly no warnings.
The basic syntax for a check is:
sudo tripwire --check

You should see a report output to your screen specifying that there were no errors or changes found on your system.

Once this is complete, you can be fairly confident that your configuration is correct. We should clean up our files a bit to remove sensitive information from our system.
We can delete the test_results file that we created:
sudo rm /etc/tripwire/test_results
Another thing that we can do is remove the actual plain text configuration files. We can do this safely because they can be generated at-will from the encrypted files with our password.

All we have to do to regenerate the plain text file is pass the encrypted file to twadmin, in much the same way that we did to generate the encrypted version. We just pipe it into a plain text file again:
sudo sh -c 'twadmin --print-polfile > /etc/tripwire/twpol.txt'
Test this now by moving the text version to a backup location and then recreate it:
sudo mv /etc/tripwire/twpol.txt /etc/tripwire/twpol.txt.bak
sudo sh -c 'twadmin --print-polfile > /etc/tripwire/twpol.txt'
If it worked correctly, you can safely remove the plain text files now:
sudo rm /etc/tripwire/twpol.txt
sudo rm /etc/tripwire/twpol.txt.bak

Set Up Email Notifications (Optional step, likely not needed for competition)

We will configure tripwire to run every day and also implement automatic notifications. During the process, we can test how to update the database when we make changes to our system.

We will use the mail command to mail our notifications to our email address. This is not installed on our system currently, so we will have to download it from the repositories.

This gives us a great opportunity to see how tripwire reacts to changes in the system.
Install the files like this:
sudo apt-get install mailutils

Now that we have that command installed, let's do a test of our system's ability to mail out a tripwire report. This report will have warnings and changes too, since we just installed new software without telling tripwire:
sudo tripwire --check | mail -s "Tripwire report for `uname -n`" your_email@domain.com

You should receive a report shortly in your email with details about the new mail software you just installed to send the message! This is good. It means that tripwire is picking up changes in the filesystem and that our mail software is working as well.

We should now "okay" the software changes we made by doing an interactive check to update the database.
We can do this by typing:
sudo tripwire --check --interactive

This will run the same tests as normal, but at the end, instead of outputting the report to the screen, it is copied into a text file and opened with the default editor.

This report goes into quite a lot of detail about each file that changed. In fact, on my machine, the report generated was 2,275 lines long. This amount of information is extremely helpful in the event of a real security problem, but in our case, it's generally probably not too interesting for the most part.

The important part is near the top. After some introductory information, you should see some lines with check boxes for each of the added or modified files:

```
Rule Name: Other binaries (/usr/sbin)
Severity Level: 66
-------------------------------------------------------------------------------

Remove the "x" from the adjacent box to prevent updating the database
with the new values for this object.

Added:
[x] "/usr/sbin/maidag"

Modified:
[x] "/usr/sbin"
. . .
```

These check boxes indicate that you want to update the database to allow these changes. You should search for every box that has an "x" in it and verify that those are changes that you made or are okay with.

If you are not okay with a change, you can remove the "x" from the box and that file will not be updated in the database. This will cause this file to still flag tripwire on the next run.

After you have decided on which file changes are okay, you can save and close the file.

At this point, it will ask for your local passphrase so that tripwire can update its database files.

If we accepted all of the changes, if we re-run this command, the report should be much shorter now and list no changes.

Automate Tripwire with Cron (Optional step, likely not needed for competition)

Now that we have verified that all of this functionality works manually, we can set up a cron job to execute a tripwire check every morning.
We will be using root's crontab, because edits to the system cronjob can get wiped out with system updates.

Check to see if root already has a crontab by issuing this command:
sudo crontab -l
If a crontab is present, you should pipe it into a file to back it up:
sudo sh -c 'crontab -l > crontab.bad'
Afterwards, we can edit the crontab by typing:
sudo crontab -e
If this is your first time running crontab, it will ask you which editor you wish to use. If you don't have a preference for another editor, nano is typically a safe choice.
Afterwards, you will be taken to a file where we can automate tripwire. Since we will be running tripwire daily, we only need to decide what time we want it to run.
Typically, services are run in non-peak times to not disrupt busy hours.

The format we need to use is min hour * * * command. The command that we want to use is the same one we used to mail our report before. We don't need to use sudo since this is going to be run as root.

To have tripwire run at 3:30am every day, we can place a line like this in our file:
30 3 * * * /usr/sbin/tripwire --check | mail -s "Tripwire report for `uname -n`" your_email@domain.com
You can adjust this to your preference.

Conclusion

You should now have an automated intrusion detection system that sends you reports regarding changes on your filesystem. You should review the emailed reports regularly and take action where there are changes detected, either in updating the tripwire database to okay the changes, or investigating suspicious activity.

Audit the Box w/ OpenVAS

Install OpenVAS

By Justin Ellingwood

Introduction

An important aspect of server security is being proactive about security screening. If you expose any services to the internet, penetration testing is essential to ensure that you are not vulnerable to known threats.

The Open Vulnerability Assessment System, known more commonly as **OpenVAS**, is a suite of tools that work together to run tests against client computers using a database of known exploits and weaknesses. The goal is to learn about how well your servers are guarded against known attack vectors.

In this guide, we will install the OpenVAS suite on an Ubuntu 12.04 VPS. We can then use this system to scan itself and other servers.

Add the OpenVAS PPA and Install Software

Although there are some OpenVAS components in Ubuntu's default repositories, we will use a PPA that maintains updated versions of the packages.

To begin with, we need to install the python-software-properties package, which will allow us to work with PPAs easily.

sudo apt-get update
sudo apt-get install python-software-properties

We can then add the newest stable version to our system:

sudo add-apt-repository ppa:openvas/openvas6

We need to rebuild the apt database to gather information about the packages available through our new PPA. Afterwards, we can install the needed software:

sudo apt-get update
sudo apt-get install openvas-manager openvas-scanner openvas-administrator openvas-cli greenbone-security-assistant sqlite3 xsltproc texlive-latex-base texlive-latex-extra texlive-latex-recommended htmldoc alien rpm nsis fakeroot

This will download and install the components that are necessary to get us started.

Initial Configuration

We can create SSL certificates for OpenVAS using a wrapper utility that is included by default. We need to call this with administrative privileges so that it can be placed in a restricted portion of the file system.

sudo openvas-mkcert

You will be asked a number of questions that will help you build a certificate file for use with this server.

Most of the questions, you can just type ENTER to accept the default values. This is mainly for your own use, so input the values you would like to use.

Next, we will create another certificate. This time, we will create a client certificate for a user named "om" this stands for OpenVAS Manager.

We don't need any specific information for the client portion, so we will tell it to configure things automatically and install the certificates in the needed places:

sudo openvas-mkcert-client -n om -i

Build the Database Information

Now that we have our certificates installed, we can begin building our database so that our local tools are aware of different kinds of threats and vulnerabilities.

Update the network vulnerability tests database by issuing this command:
sudo openvas-nvt-sync

This will download the latest definitions to your local machine.

To continue we'll need to stop the manager and scanner applications so that we can call the commands without a conflict momentarily.

Stop both of these services by typing:
sudo service openvas-manager stop
sudo service openvas-scanner stop

Now, we can start the scanner application without the parameters found in the init file that are usually called.

During this first run, OpenVAS will need to download and sync a lot of data. This will take quite awhile:
sudo openvassd

Once this is finished, you'll have to rebuild the database generated by the scanner by typing:
sudo openvasmd --rebuild

Next, we will download and update our security content automation protocol data. This is known as "SCAP" data. This is another database that OpenVAS checks against for our security tests.
sudo openvas-scapdata-sync

This will be another long wait. It downloads some general files and then updates them in the database.
We will then run a similar sync operation for the cert data:
sudo openvas-certdata-sync

Running this command for the first time, you might see some errors. They may look something like this:
Error: no such table: meta

This is because the Ubuntu package is actually missing some files that are packaged in some other versions.
We can get these from an RPM package for the manager component.

Type this to download it to your home directory:
cd
wget http://www6.atomicorp.com/channels/atomic/fedora/18/i386/RPMS/openvas-manager-4.0.2-11.fc18.art.i686.rpm

Now that we have the file downloaded, we can extract and expand the directory structure present within the RPM. We can do this by typing:
rpm2cpio openvas* | cpio -div

We will create a directory for our new files in a place where OpenVAS will find them. We will then move the files into that directory:
sudo mkdir /usr/share/openvas/cert
sudo cp ./usr/share/openvas/cert/* /usr/share/openvas/cert

Now, we can safely run the cert syncing command again, and it should complete as expected this time:
sudo openvas-certdata-sync

Afterwards, we can delete the extracted RPM data and directories from our home directory:
rm -rf ~/openvas* ~/usr ~/etc

Set Up OpenVAS User and Ports

To sign into our service, we will need a user. We can create one with the OpenVAS administrator component.
Here, we will create a user called "admin" with the role of administrator. You will be asked to provide a password to use for the new account:
sudo openvasad -c add_user -n admin -r Admin

You will be informed that the user has been granted unlimited access.

Next, we need to change the way one of our components starts up. The Greenbone Security Assistant component is a web-based interface to the tools we have installed.

By default, the interface is only accessible from the local computer. Since we are installing the OpenVAS suite on a remote server, we won't be able to access the web interface with these settings. We need to make it accessible from the internet.

Open the following file with root privileges in your preferred text editor:
sudo nano /etc/default/greenbone-security-assistant

Near the top, you should see a parameter that specifies the address that the web interface will listen on.

We need to change the value from 127.0.0.1 to the public IP address of your VPS. This will let it listen to connections from the internet, and we will be able to connect:
GSA_ADDRESS=your_server_IP_address

Save and close the file when you have made the above modification.

Start Up the Services

We will now start up the services that we have been configuring. Most of them are already running in some capacity, but we will have to restart them to make sure that they use the new information we have been gathering.

Start by killing all of the running OpenVAS scanner processes:
sudo killall openvassd

It can take up to 15 or 20 seconds for the process to actually be killed. You can check if there are processes still running by issuing:
ps aux | grep openvassd | grep -v grep

If anything is returned, then your processes have not finished yet and you should continue to wait.

Once the process exits completely, you can begin starting all of your services again:
sudo service openvas-scanner start
sudo service openvas-manager start
sudo service openvas-administrator restart
sudo service greenbone-security-assistant restart

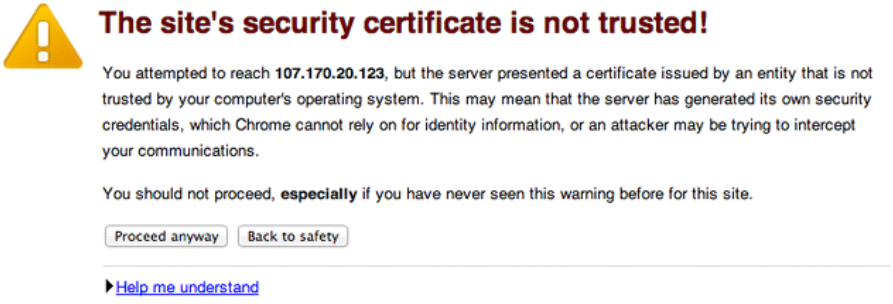
Each of these may take a bit of time to start up.

Access the Web Interface and Run Some Tests

Once the services have all started, you use your web browser to access the Greenbone Security Assistant web interface.

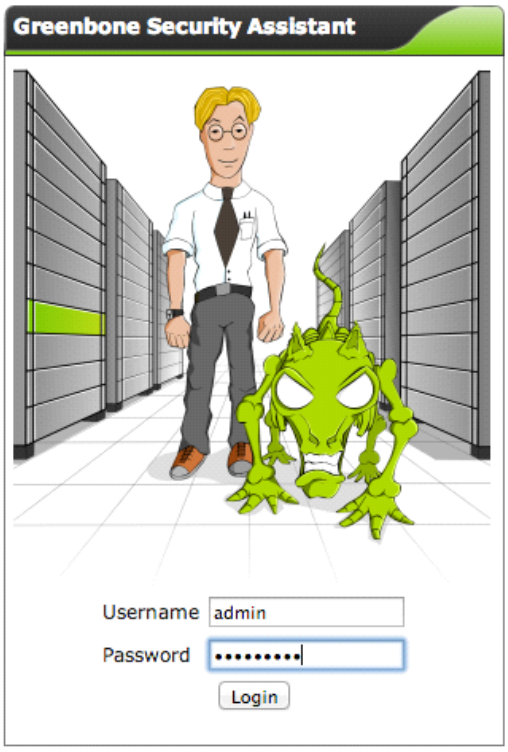
To access this, you **must** precede the address of your server with https://. You then enter either the domain name or IP address of your server followed by :9392.
https://server_domain_or_IP:9392

You will be presented with a scary looking warning screen telling you that the certificate is not signed by someone that your browser trusts by default:



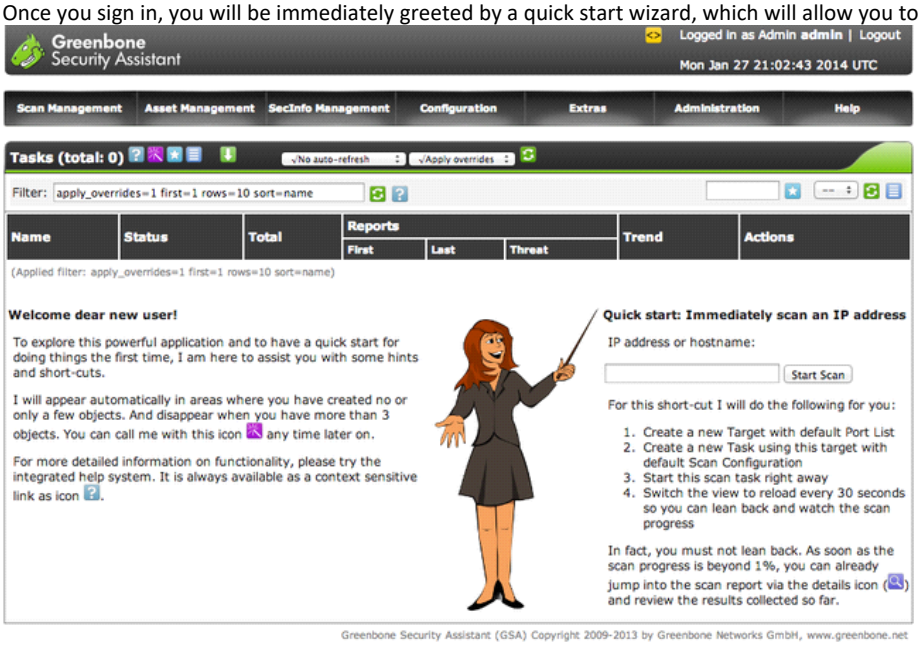
This is expected and not a problem. You should click on the "Proceed anyway" button to continue.

Next, you will be presented with the login screen:



You will need to enter the username and password you configured earlier. For this guide, the username was "admin".

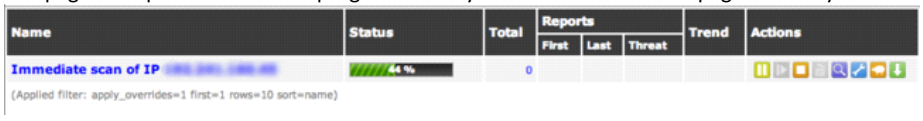
Once you sign in, you will be immediately greeted by a quick start wizard, which will allow you to run a default scan against a target computer right away:



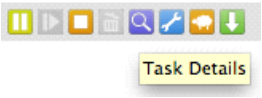
A good choice is to run against another server that you own. It is important to not run these scans against targets that are not under your control, because they may look like potential attacks to other users.

Enter the IP address of the computer you wish to test against and click the "Start Scan" button to begin.

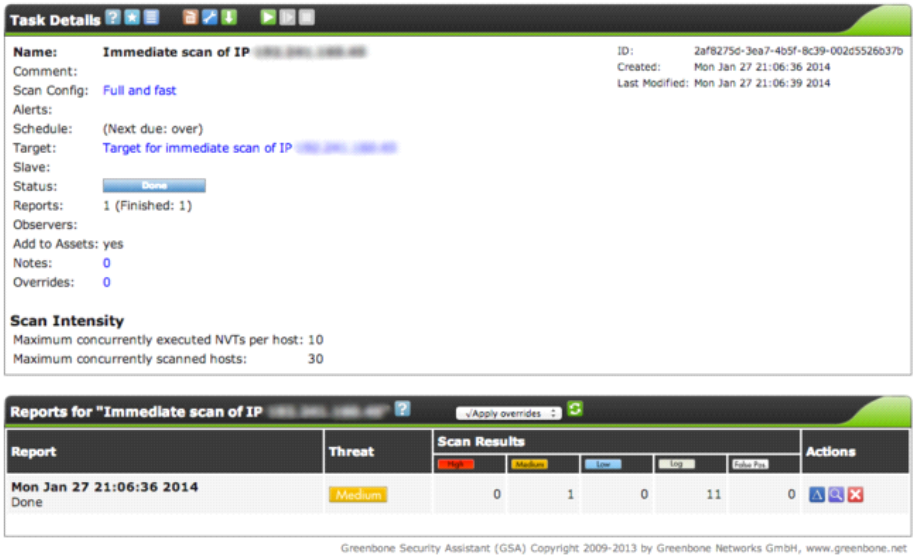
The page will update as the scan progresses and you can also refresh the page manually to track the progress:



When the scan is complete (or even before if you want to review the information as it is coming in), you can click on the purple magnifying glass icon to see the results of the scan. It is normal for the scan to rest on 98% for a while before completing:



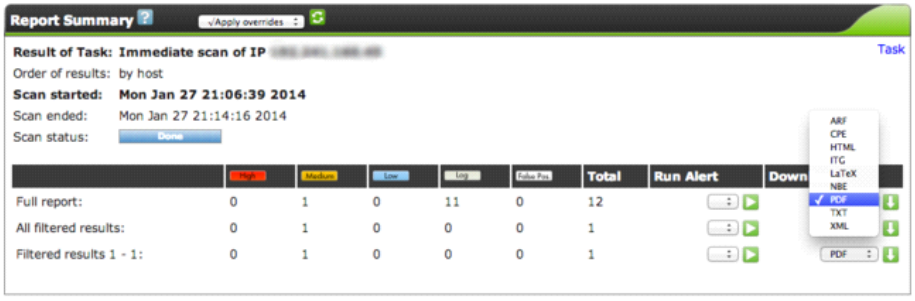
You will be taken to an overview of the scan results. Note that the immediate scan that we have completed is not the most in-depth scan we have available.



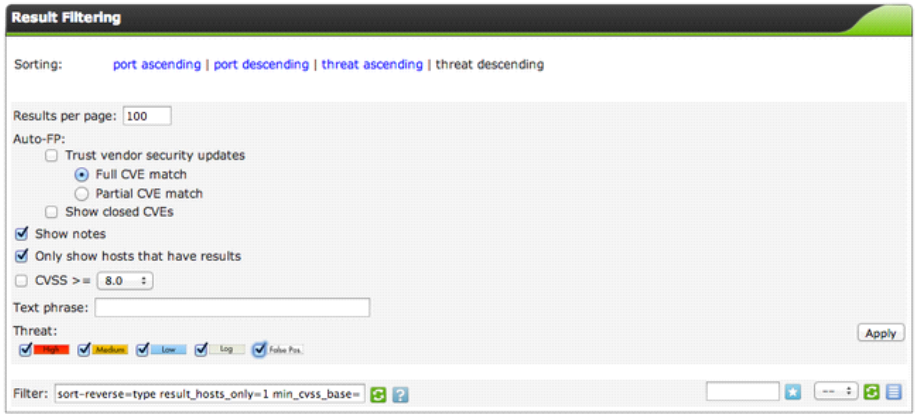
At the bottom, you can see the report that OpenVAS created telling us of potential vulnerabilities in the system we scanned. We can see that the "Threat" level has been categorized as "Medium".

This means that at least one vulnerability has been found in the system of the rating of "medium". We can find out more by clicking on the magnifying glass again.

This will take us to a full report of the findings. In the top portion, you have the option to download the results in various formats:



In the middle section, we can filter the results. By default, the interface will only show threats marked "high" or "medium". On your first time through, you should probably check all of the boxes under the "Threat" category. Click "Apply" to implement this:



The bottom section tells us about the specific items that were found. If you selected all of the boxes above, you will see some informational messages about open ports and similar findings.

The threats will be color-coded to match their button color. For instance, this is our medium threat:

Medium (CVSS: 2.6) general/tcp
NVT: TCP timestamps (OID: 1.3.6.1.4.1.25623.1.0.80091)

Result:

```
It was detected that the host implements RFC1323.

The following timestamps were retrieved with a delay of 1 seconds in-between:
Paket 1: 3264915
Paket 2: 3265185
```

Impact

A side effect of this feature is that the uptime of the remote host can sometimes be computed.

Solution

To disable TCP timestamps on linux add the line 'net.ipv4.tcp_timestamps = 0' to /etc/sysctl.conf. Execute 'sysctl -p' to apply the settings at runtime.

To disable TCP timestamps on Windows execute 'netsh int tcp set global timestamps=disabled'

Starting with Windows Server 2008 and Vista, the timestamp can not be completely disabled.

The default behavior of the TCP/IP stack on this Systems is, to not use the Timestamp options when initiating TCP connections, but use them if the TCP peer that is initiating communication includes them in their synchronize (SYN) segment.

See also: http://www.microsoft.com/en-us/download/details.aspx?id=9152

Vulnerability Insight

The remote host implements TCP timestamps, as defined by RFC1323.

Vulnerability Detection Method

Special IP packets are forged and sent with a little delay in between to the target IP. The responses are searched for a timestamps. If found, the timestamps are reported.

References

Other: http://www.ietf.org/rfc/rfc1323.txt

This warning tells us that our target responds to timestamp requests. These requests can let an attacker know how long the host has been online continuously. This could let an attacker know that the host is vulnerable to any recent exploits.

As you can see, the report also includes information about how to address the issue.

Conclusion

You should now have a fully functional OpenVAS server set up to scan your hosts. This can help you spot vulnerabilities and highlight areas to focus on when you are tightening up security.

We have only shown a bare minimum of the functionality of the OpenVAS security suite. Among other tasks, you can easily schedule scans, automatically generate reports, and email alerts when certain threat levels are generated. Explore the Greenbone Security Assistant interface and take advantage of the great built-in help system to learn more about your options.

Monitor the Network

Install iptraf

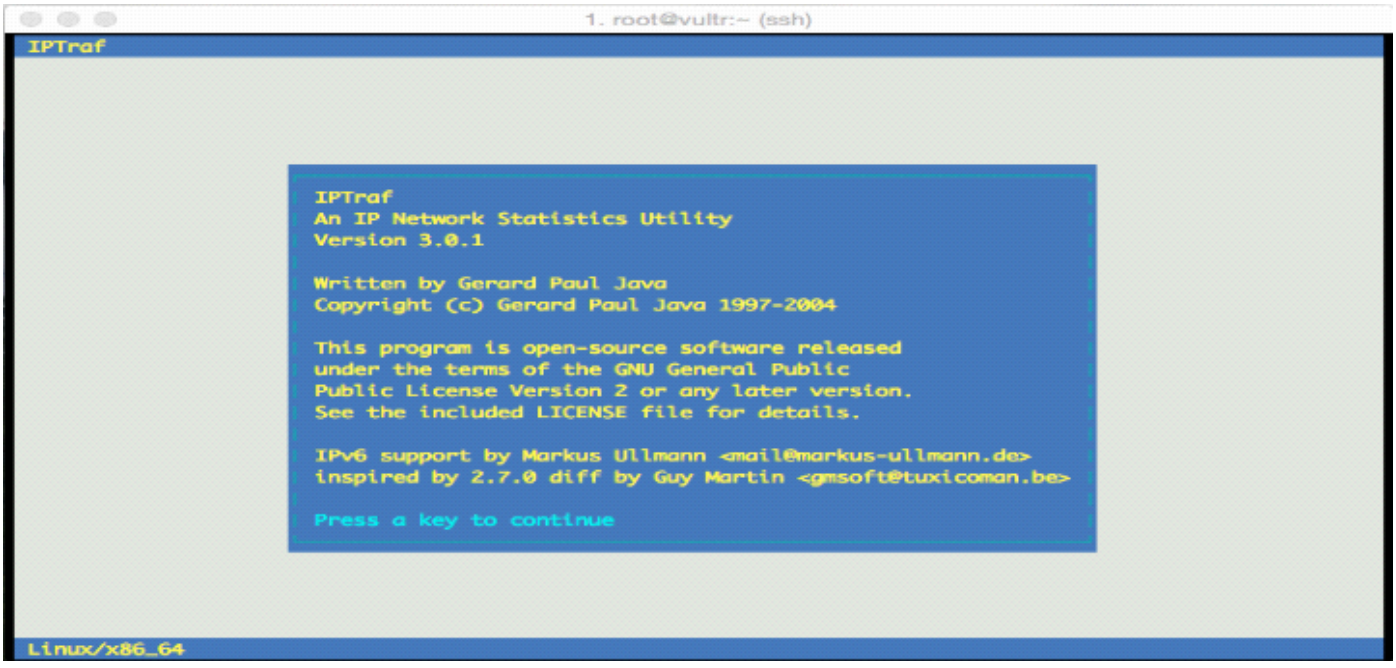
iptraf is a network analysis tool for Linux that recognizes various protocols from TCP and UDP to ICMP, ARP, IGP, and more.

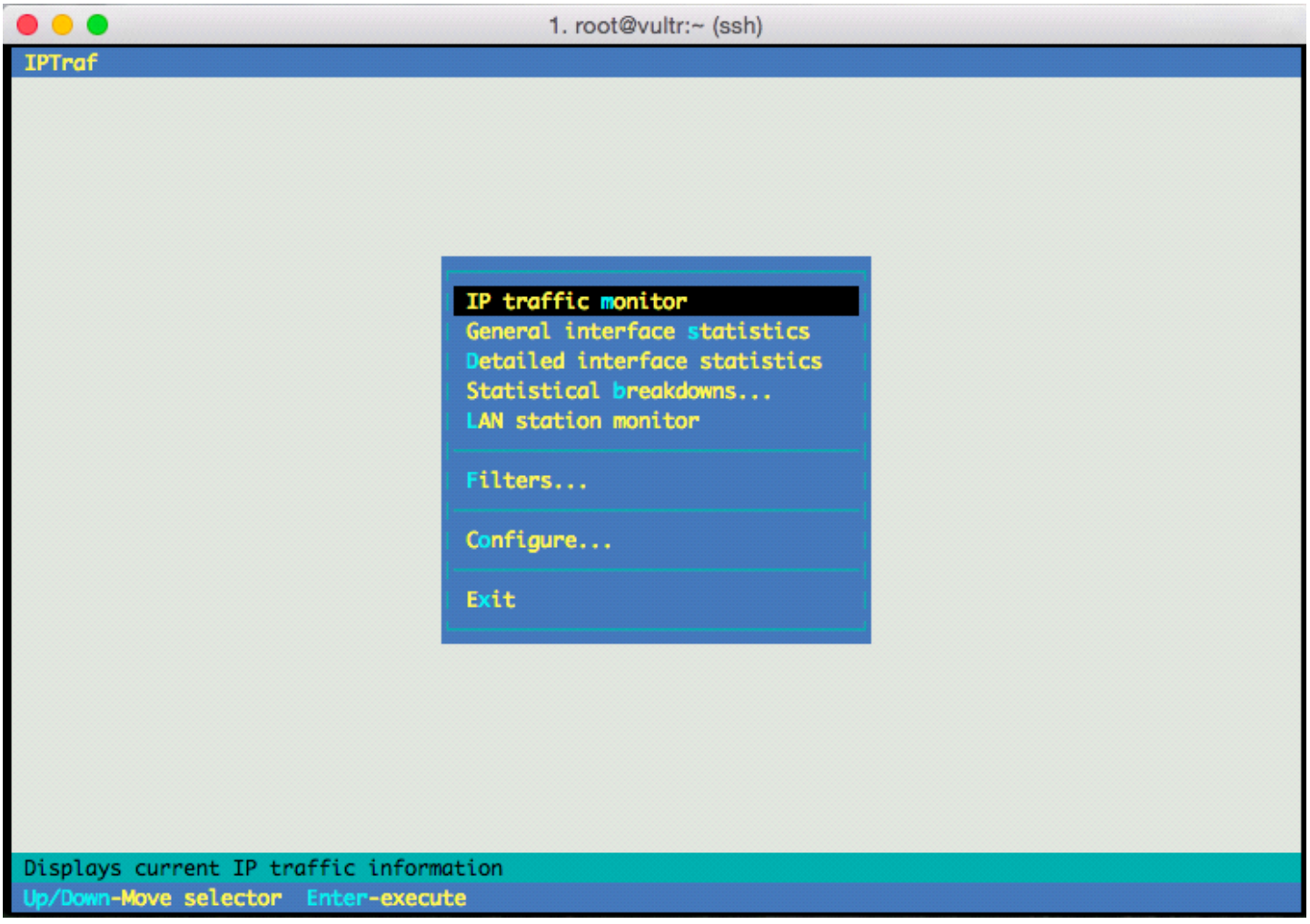
First. install iptraf

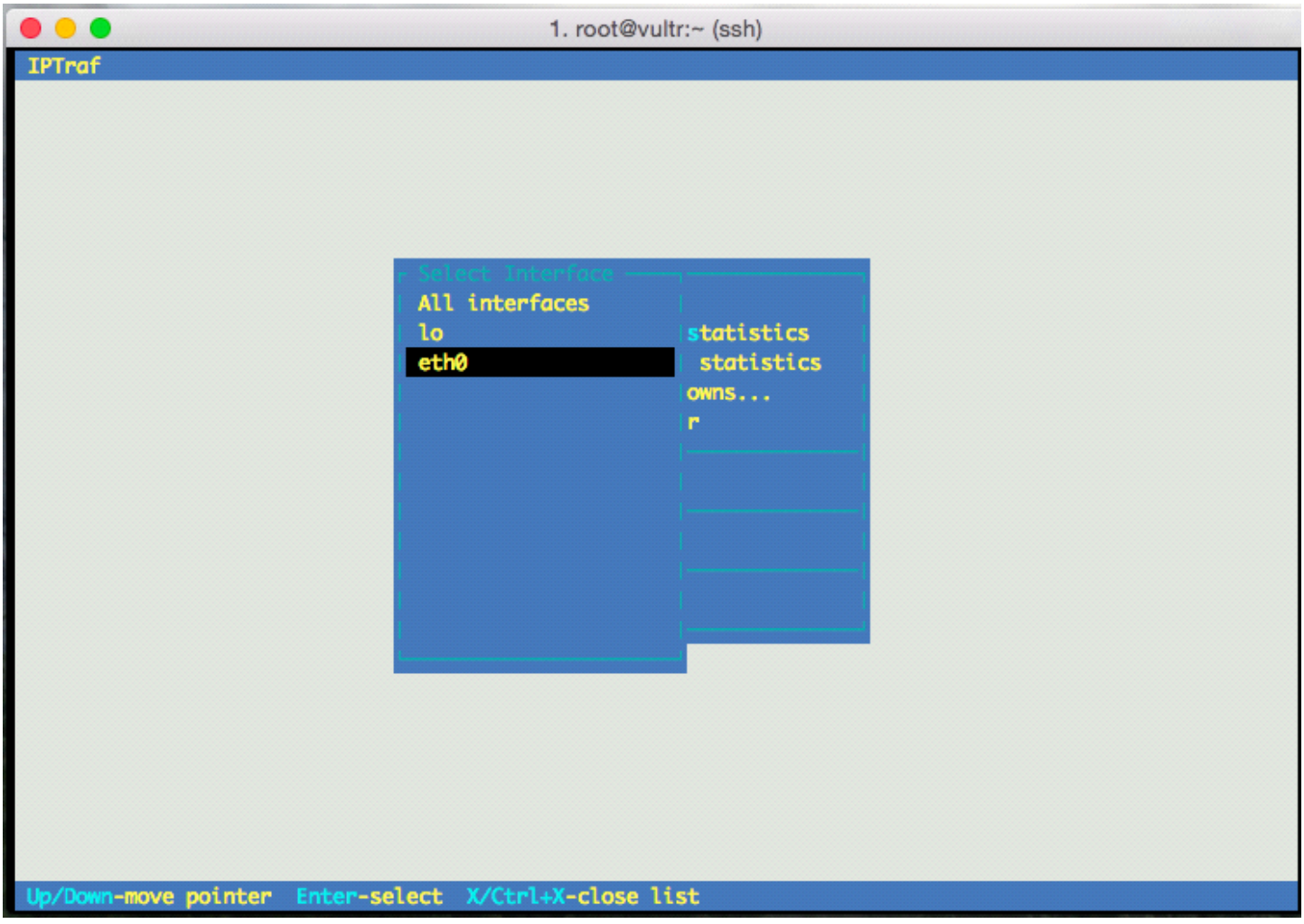
sudo apt-get install iptraf

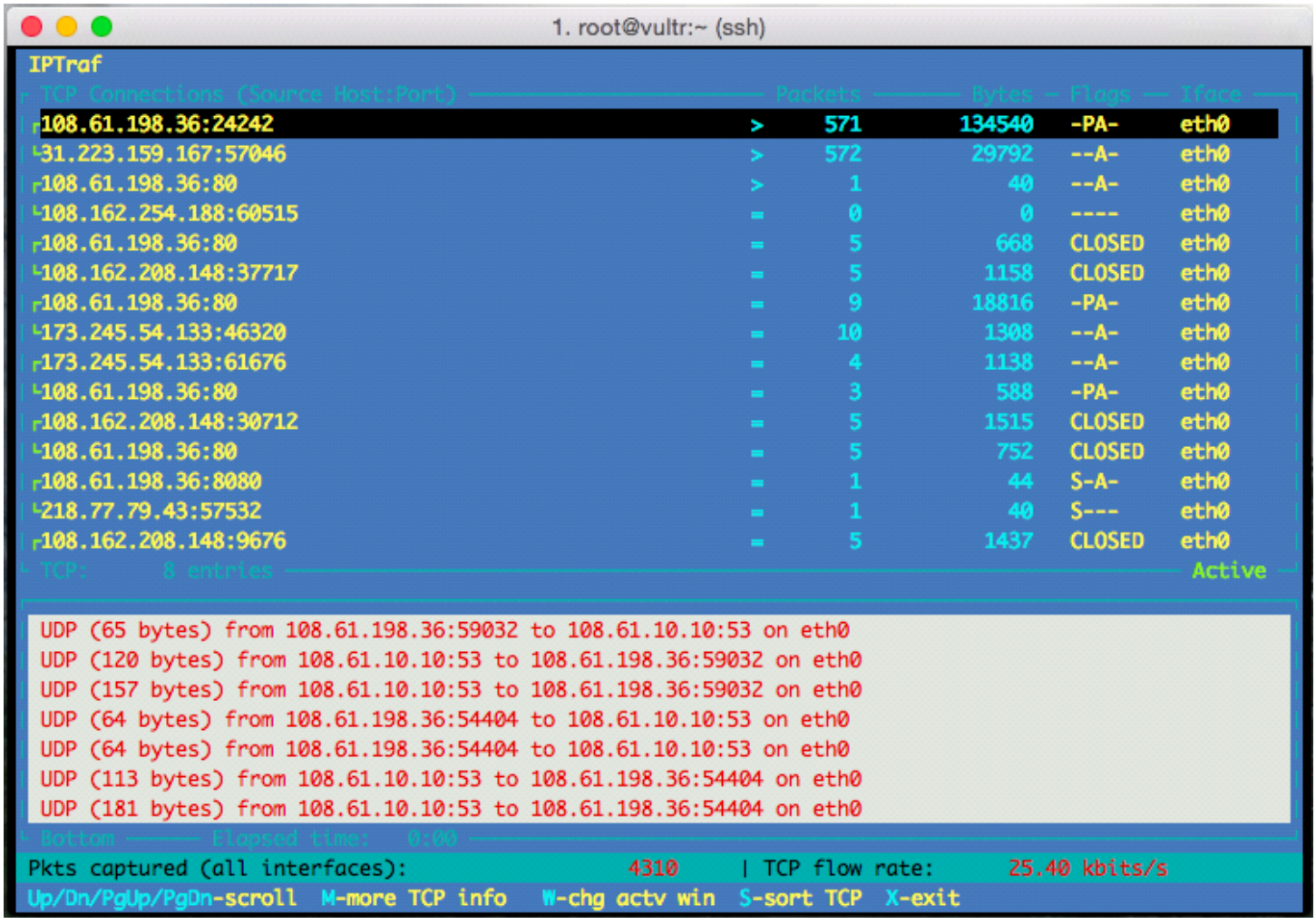
Then launch iptraf in interactive mode by simply typing

iptraf




1. root@vultr:~ (ssh)
IPTraf
IP traffic monitor
General interface statistics
Detailed interface statistics
Statistical breakdowns...
LAN station monitor
Filters...
Configure...
Exit
Displays current IP traffic information
Up/Down-Move selector Enter-execute


1. root@vultr:~ (ssh)
IPTraf
Select Interface
All interfaces
lo
eth0
statistics
statistics
owns...
r
Up/Down-move pointer Enter-select X/Ctrl+X-close list



From <http://lintut.com/install-iptraf-tcpudp-network-monitoring-utility-on-linux/>

Chapter 7. HOUSEKEEPING

Automate It

Install Ubuntu Secure Server Script

WARNING: IF YOU USED CHAPTER %00 SCRIPTS, DO NOT USE THIS TOOL!

First, install zenity using *apt-get*.

sudo apt-get install zenity

Second, wget the Ubuntu Server Secure script.

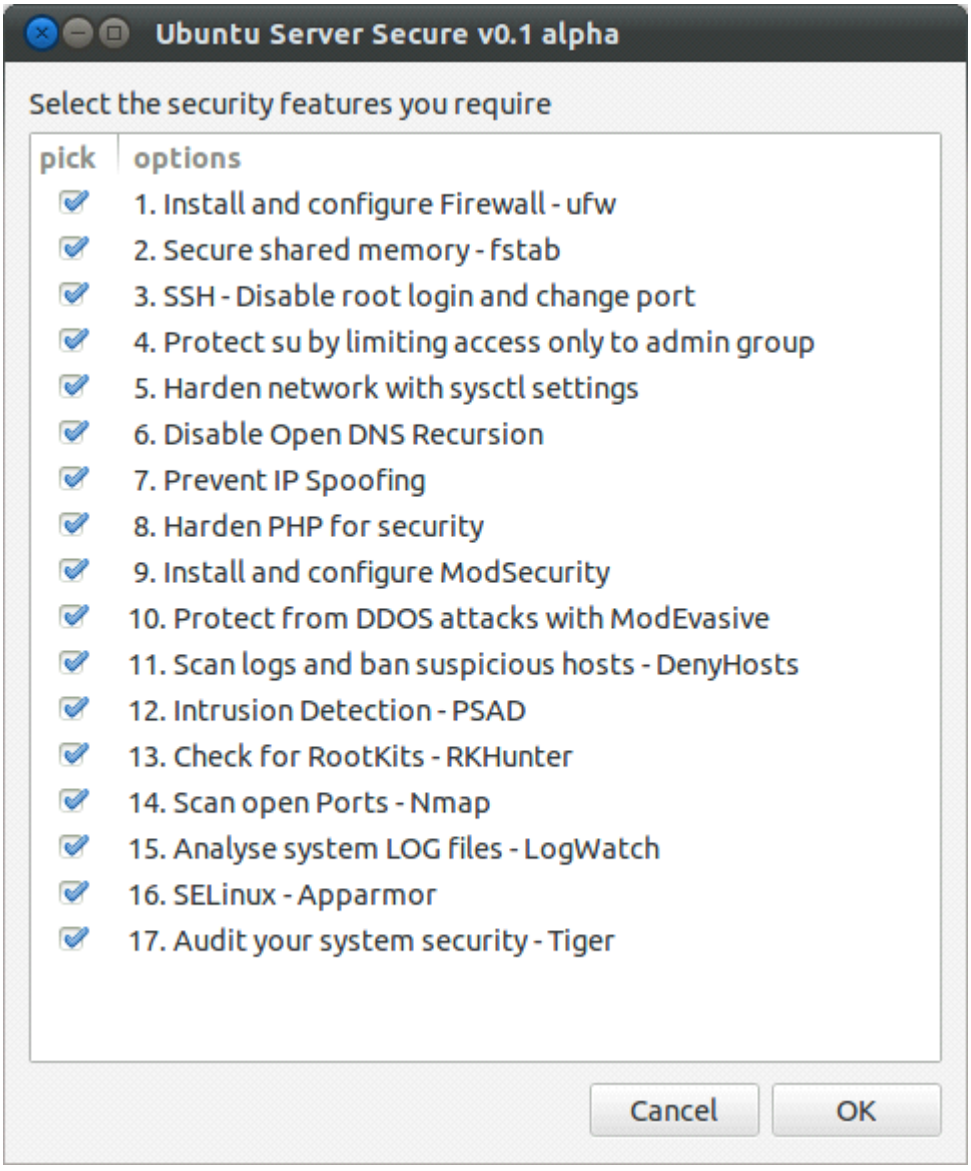
wget https://www.thefanclub.co.za/sites/all/modules/pubdlcnt/pubdlcnt.php?file=https://www.thefanclub.co.za/sites/default/files/public/downloads/ubuntu-server-secure.tar.gz

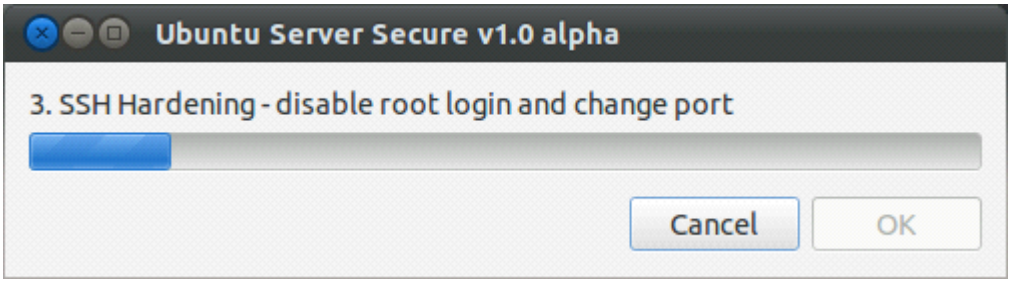
Then, extract the archive using *tar*.

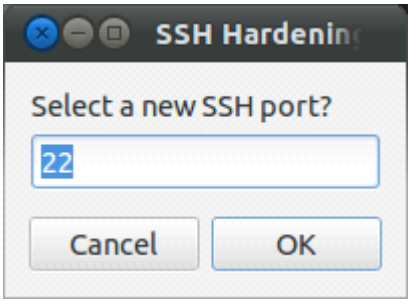
tar -zxvf ubuntu-server-secure.tar.gz
cd ubuntu-server-secure
sudo chmod +x ubuntu-server-secure.sh

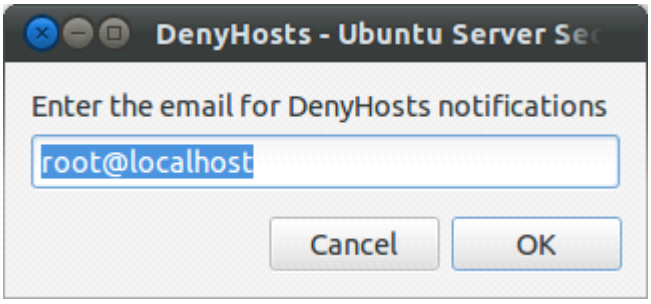
Finally, use the *gksudo* command to launch the GUI.

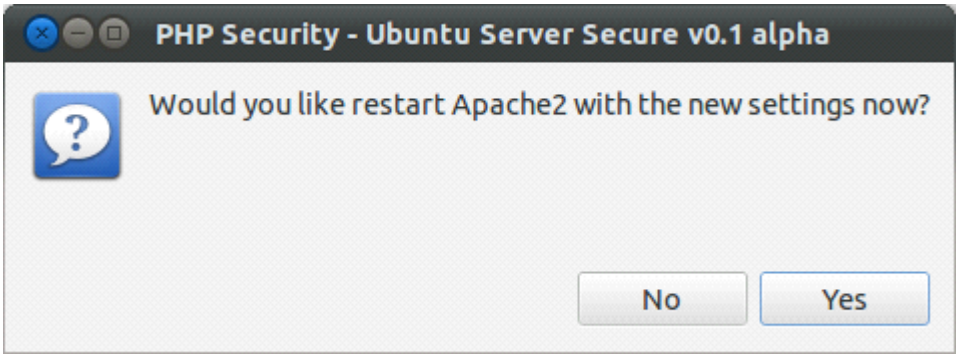
gksudo sh ubuntu-server-secure.sh

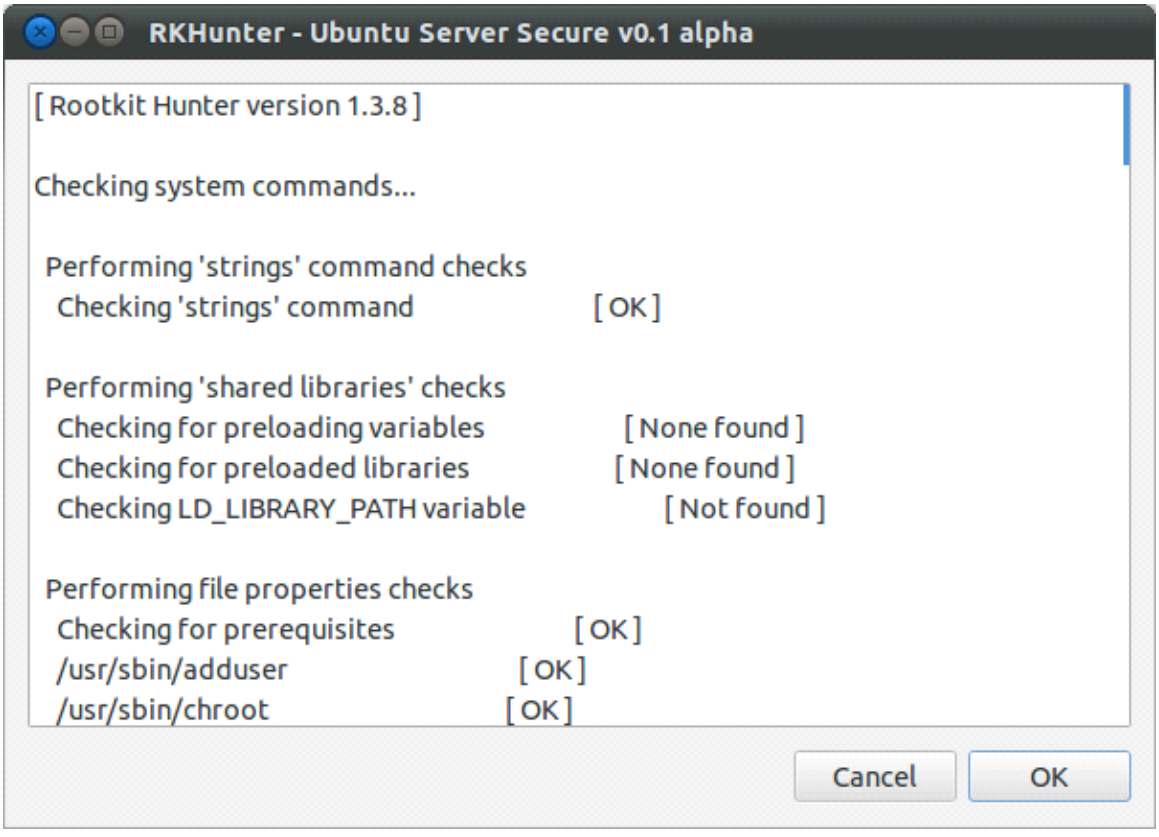

Ubuntu Server Secure v0.1 alpha
Select the security features you require
pick
options
1. Install and configure Firewall - ufw
2. Secure shared memory - fstab
3. SSH - Disable root login and change port
4. Protect su by limiting access only to admin group
5. Harden network with sysctl settings
6. Disable Open DNS Recursion
7. Prevent IP Spoofing
8. Harden PHP for security
9. Install and configure ModSecurity
10. Protect from DDOS attacks with ModEvasive
11. Scan logs and ban suspicious hosts - DenyHosts
12. Intrusion Detection - PSAD
13. Check for RootKits - RKHunter
14. Scan open Ports - Nmap
15. Analyse system LOG files - LogWatch
16. SELinux - Apparmor
17. Audit your system security - Tiger
Cancel
OK


Ubuntu Server Secure v1.0 alpha
3. SSH Hardening - disable root login and change port
Cancel
OK


SSH Hardenin
Select a new SSH port?
22
Cancel
OK







From <https://www.thefanclub.co.za/how-to/how-secure-ubuntu-1204-lts-server-part-2-gui-installer-script>

Example Scenarios & Solutions

Scenario #1

This server needs to be cleaned up so that it can be used for another purpose. You are assigned the task of cleaning up and hardening the server, including the removal of current services. Login capability should be limited to the Argus account. Argus should remain the only account that is allowed root privileges.

cp-ub10-01
Solution

1. Change argus's password from Windows12 to ?Defen08
   a. Type:
      1) passwd argus
      2) New password: ?Defen08

2. Authentication
   a. Install cracklib
      1) apt-get install libpam-cracklib --force-yes -y

   b. Setup failed login attempts
      1) gedit /etc/pam.d/common-auth
      2) auth optional pam_tally.so deny=5 unlock_time=900 onerr=fail audit even_deny_root_account silent

   c. Minimum Password Length & Composition
      1) gedit /etc/pam.d/common-password

      2) password requisite pam_cracklib.so retry=3 minlen=8 difok=3 reject_username minclass=3 maxrepeat=2 dcredit=1 ucredit=1 lcredit=1 ocredit=1

   d. Password History
      1) password requisite pam_pwhistory.so use_authtok remember=24 enforce_for_root

   e. Hashing Algorithm
      1) password [success=1 default=ignore] pam_unix.so obscure use_authtok sha512 shadow

3. Change each user's password, type:
   a. passwd username
   b. New password: ?Defen08

4. Lock all users except argus
   a. passwd –l username

5. Limit root access
   a. gedit /etc/group
   b. Remove all users from the admin group except argus

6. System Patching
   a. apt-get update
   b. apt-get dist-update –y

a. apt-get update
b. apt-get dist-update –y

7. Netcat Backdoor
   a. Search for a process running nc
      1) ps –ef | grep nc
   b. Search for the program that started netcat
      1) grep –l '/bin/nc' /etc/rc*
   c. Remove the line that starts netcat
      1) gedit /etc/rc.local

8. Scheduled job
   a. rm /var/spool/cron/crontabs/root

9. Service removal
   a. List running services
      1) service - -status-all
   b. Remove all services
      1) apt-get - -purge autoremove servicename

Scenario #2

This server has been used for many purposes and needs to be seriously cleaned up. The only current service that should remain is Telnet. You are assigned the task of cleaning up and hardening the server, including the removal of unnecessary services. Login capability should be limited to Argus and Sierra. These accounts will be the only administrators for this server.

cp-ub10-02
Solution

1. Change argus's password from Windows12 to ?Defen08
   a. Type:
      1) passwd argus
      2) New password: ?Defen08

2. Authentication
   a. Install cracklib
      1) apt-get install libpam-cracklib --force-yes -y

   b. Setup failed login attempts
      1) gedit /etc/pam.d/common-auth
      2) auth optional pam_tally.so deny=5 unlock_time=900 onerr=fail audit even_deny_root_account silent

   c. Minimum Password Length & Composition
      1) gedit /etc/pam.d/common-password

      2) password requisite pam_cracklib.so retry=3 minlen=8 difok=3 reject_username minclass=3 maxrepeat=2 dcredit=1 ucredit=1 lcredit=1 ocredit=1

   d. Password History
      1) password requisite pam_pwhistory.so use_authtok remember=24 enforce_for_root

   e. Hashing Algorithm
      1) password [success=1 default=ignore] pam_unix.so obscure use_authtok sha512 shadow

3. Setup Password Policy
   a. gedit /etc/login.defs
   b. Change the PASS . . line to reflect the following:
      1) PASS_MIN_DAYS 7
      2) PASS_MAX_DAYS 90
      3) PASS_WARN_AGE 14

4. Change the Password Policy for existing users
   a. chage –m 7 –M 90 –I 10 –W 14 username

5. Change each user's password, type:
   a. passwd username
   b. New password: ?Defen08

6. Lock Larry, Curly and Moe
   a. passwd –l username

7. Remove argus's password from the GECOS field.
   a. gedit /etc/passwd

8. Limit root access
   a. gedit /etc/group
   b. Remove curly from the admin group

9. System Patching
   a. apt-get update
   b. apt-get dist-update –y

10. Netcat Backdoor
    a. Search for a process running nc
       1) ps –ef | grep nc
    b. Search for the program that started netcat
       1) grep –l '/bin/nc' /etc/rc*
    c. Remove the line that starts netcat
       1) gedit /etc/rc.local

11. Scheduled job
    a. rm /var/spool/cron/crontabs/root

12. Telnet service
    a. Fix the telnet config file
       1) gedit /etc/xinetd.d/telnet
       2) Remove the following line:
          a) server_args = -L /usr/local/bin/autologin
       3) Added the following line:
          a) only_from = 127.0.0.1 192. 168.1.0/24
    b. Remove banners
       1) gedit /etc/issue.net
          a) Remove Ubuntu line
       2) gedit /etc/update-motd.d/00-header
          a) Comment out all lines
       3) gedit /etc/update-motd.d/10-help-test
          a) Comment out all lines

13. Service removal
    a. List running services
       1) service - -status-all
    b. Remove all services except telnet
       1) apt-get - -purge autoremove servicename

Chapter %00
EOF