**TIPS:**

1. Don’t use any of their terminals. During the practice sessions with ryan, we discovered there are hidden shells within launching a terminal window.
   1. $ exit
   2. Echo $$
   3. Echo $PS1
      1. If you type in exit and it does not close the entire terminal window. It was rigged. Do not trust it until you know it's safe.
2. Don't use any of their text editors, others have reported text editors being rigged to
3. Grep for plain default text password in /etc / . this
   1. At the beginning to see where its at
   2. Change password
   3. Grep again to see if it's changed.
4. deleting entire files. Download emacs and run that GNU master race all night and day.
5. Check who is logged in and the tty there are in. try to write this down. Or output it with a time stamp. That way you can always check if anything else has logged in.
6. See who was last logged in and what happened: $ last and output that to a file
7. Naming conventions for saving config files and audit reports (auditing refers to any checks that return valuable or historic information):
   1. Config File: service\_path-to-config.hist  
       eg: apache\_etc-httpd-conf-httpd.conf.hist
   2. Audit Info: audit-tool-name\_reason.info  
       eg: who\_list-current-users.info

# Search and Mitigate

**Check cron jobs and daemons**

$ crontab -e

*BSD Unix*

$ emacs /var/cron/tabs/

*Solaris, HP-UX, Debian, Ubuntu*

$ emacs /var/spool/cron/crontabs/

*AIX, Red Hat Linux, CentOS, Fedora*

$ emacs /var/spool/cron/

$ emacs /etc/crontab

file, and /etc/cron.\*/ directories.

It also checks the /var/spool/cron/

**Check RC Scripts (init.d/ openRc/ sysVinit)**

#https://fedoraproject.org/wiki/SysVinit\_to\_Systemd\_Cheatsheet

**# Will Display all running/startup services**

$chkconfig --list

**# Check system start up on runlevel 5(replace with graphical runlevel)**

$chkconfig --list | grep 5:on

**Check RC Scripts (Systmd/Systemctl)**

**# Tested on Arch Linux running systemctl**

**# Will display start up services**

$ systemctl list-unit-files --type=service (or) $ls /etc/systemd/system/\*.wants/

**# Check system start up on runlevel 5(replace with graphical runlevel)**

$systemctl list-dependencies graphical.target

**Check Bashrc and bash\_profile**

*#Make sure to check /root/.bashrc*

$ emacs ~/.bashrc

$ emcas ~/.bash\_profile

*# now that everything is “safe,” change the password to root*

**Change root and other admins password**

$ passwd root

**Backup /etc and APACHE\_ROOT\_DIR.**

*#Will time stamp later*

$ mkdir BackupDir

$ tar -cvpzf /BackupDirectory/etc\_backup\_one.tar.gz /etc

$ tar -cvpzf /BackupDirectory/apache\_backup\_one.tar.gz /APACHE\_ROOT\_DIR

*Could make a cron:*

30 \* \* \* \* tar -czvf ~/backups/"$(date +%F\_%R)etc.tar.gz" /etc/

30 \* \* \* \* tar -czvf ~/backups/"$(date +%F\_%R)var.tar.gz" /var/

*Backup every 30 minutes*

**Backup any mysql databases(If on your machine, Do this after you run after this checklist)**

*#EX: backing up a database named wordpress(has to be already created).* ***remotely***

$mysql -u root -p skengo < WorkingSkengoDBMay2016.sql -h127.0.0.1 --protocol=tcp

*#Dump one database*

$mysqldump database\_name > database\_name.sql

*#Dump all databases*

$mysqldump --all-databases > all\_databases.sql

**Restore Database( Might have to exists already, just create empty database)**

$mysql database\_name < database\_name.sql

**Check PATHS**

$ echo $PATH

**Check IP**

$ ifconfig $ ip addr

**Check open Ports**

$ netstat -tulpn |grep LISTEN

**Check Running programs**

$ pe -e

**Check a certain funny looking task that is running and find out more.**

Aww$ps -aux |grep vnc

**Kill running tasks**

$ kill PID\_NUMBER $ killall SERVICE\_NAME

**Get info on hardware and system release.**

\_CPUINFO=$( cat /proc/cpuinfo );

\_ISSUEINFO=$( cat /etc/issue );

\_RELEASEINFO=$( cat /etc/\*-release);

# User and Group Management

**Find users**

$ \_PASSWDF=$( cut -d: -f1 /etc/passwd) $ \_SHADOWF=$( cut -d: -f1 /etc/shadow)

**Find Groups**

$ getent group

**Check users in groups**

$ getent group USERNAME

**Check super users & disable super users. Have only 1 admin and root users.**

Make sure all these lines are commented or deleted. Eliminate everyone from having super access. And remove them from their groups.

$ emacs /etc/sudoers

*# %wheel ALL=(ALL) ALL*

*# %sudo ALL=(ALL) ALL*

*# %ALL ALL=(ALL) ALL*

**Delete user from group**

$ gpasswd -d "$\_USERSTRING4" "$\_GROUP4"

**Delete everyone from a group at once**

$ gpasswd "$\_GROUP" -M ""

**Create a new super user**

$ useradd -m -G sudo -s /bin/bash SUPERSAIYAN

**Create a user and add to a group**

$ useradd -m -G "$\_GROUP" -s /bin/bash "$\_USERSTRING"

**Add user to a group**

$ usermod -a -G "$\_GROUPTWO" "$\_USERSTRINGTWO"

# Hardening

**IPTABLES**

\*filter

# Allows all loopback (lo0) traffic and drop all traffic to 127/8 that doesn't use lo0

-A INPUT -i lo -j ACCEPT

-A INPUT ! -i lo -d 127.0.0.0/8 -j REJECT

# Accepts all established inbound connections

-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

# Allows all outbound traffic

# You could modify this to only allow certain traffic

-A OUTPUT -j ACCEPT

# Allows HTTP and HTTPS connections from anywhere (the normal ports for websites) or any #other tcp port that needs to added

-A INPUT -p tcp --dport 80 -j ACCEPT

-A INPUT -p tcp --dport 443 -j ACCEPT

# Allows SSH connections

# The --dport number is the same as in /etc/ssh/sshd\_config

-A INPUT -p tcp -m state --state NEW --dport 22 -j ACCEPT

# Now you should read up on iptables rules and consider whether ssh access

# for everyone is really desired. Most likely you will only allow access from certain IPs.

# Allow ping

# note that blocking other types of icmp packets is considered a bad idea by some

# remove -m icmp --icmp-type 8 from this line to allow all kinds of icmp:

# https://security.stackexchange.com/questions/22711

-A INPUT -p icmp -m icmp --icmp-type 8 -j ACCEPT

# log iptables denied calls (access via 'dmesg' command)

-A INPUT -m limit --limit 5/min -j LOG --log-prefix "iptables denied: " --log-level 7

# Reject all other inbound - default deny unless explicitly allowed policy:

-A INPUT -j REJECT

-A FORWARD -j REJECT

COMMIT

**SSH Lock DAT DOWN**

$ emacs /etc/ssh/sshd\_config

*Remove Root*

# Prevent root logins:  
 PermitRootLogin not

*Limit User logins*

AllowUsers alice bob

*Disable Protocol 1*

# Protocol 2,1  
 Protocol 2

*Use public and private keys.*

#First create keys on the **CLIENT.** This will create two hidden files in the ~/.ssh directory. These files are private key **id\_rsa** and #public key **id\_rsa.pub** .

$ ssh-keygen -t rsa

Now set permissions on your private key:

$ chmod 700 ~/.ssh  
$ chmod 600 ~/.ssh/id\_rsa

Copy the public key (id\_rsa.pub) to the server and install it to the authorized\_keys list:

$ cat id\_rsa.pub >> ~/.ssh/authorized\_keys

Note: once you've imported the public key, you can delete it from the server.

and finally set file permissions on the server:

$ chmod 700 ~/.ssh  
$ chmod 600 ~/.ssh/authorized\_keys

Once you've checked you can successfully login to the server using your public/private key pair, you can disable password authentication completely by adding the following setting to your **/etc/ssh/sshd\_config file:**

# Disable password authentication forcing use of keys  
PasswordAuthentication no