

## Lab 02 – Configure Basic Security Controls on CentOS

David Murillo Santiago  
Professor Zhang  
IS-3033  
3 February 2024

### INTRODUCTION

---

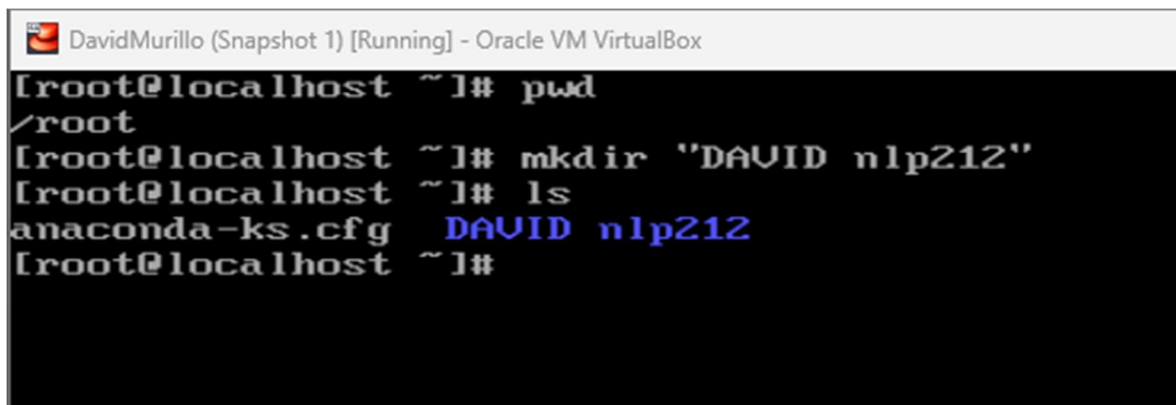
In this lab, I will configure basic security controls on CentOS by practicing Linux commands, altering boot processes for password management, hardening the GRUB bootloader, enabling SELinux, and managing firewall interfaces including iptables and firewalld.

### PROCESS

---

#### **Step 1: Create a folder in root user's home directory.**

In this step, I created a new directory within the root user's home directory. I named it after my first name and my assigned "abc123" characters.

A screenshot of a terminal window titled "DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox". The terminal shows the following commands and output:

```
[root@localhost ~]# pwd
/root
[root@localhost ~]# mkdir "DAVID nlp212"
[root@localhost ~]# ls
anaconda-ks.cfg  DAVID nlp212
[root@localhost ~]#
```

I used the 'pwd' to verify I was in home directory. Next, I used the 'mkdir' command to create a new directory.

#### **Step 2: Create a file within the new folder.**

In this step, I created a file within the newly created folder.

```
[root@localhost ~]# touch /root/DAVID\ nlp212/davidmurillo
[root@localhost ~]# ls
anaconda-ks.cfg  DAVID  nlp212
[root@localhost ~]# ls DAVID\ nlp212
davidmurillo
[root@localhost ~]#
```

*I used the touch command, followed by my desired file path (inside the new folder), to create a file named after me in the new folder.*

### **Step 3: Copy the entire folder onto a new folder.**

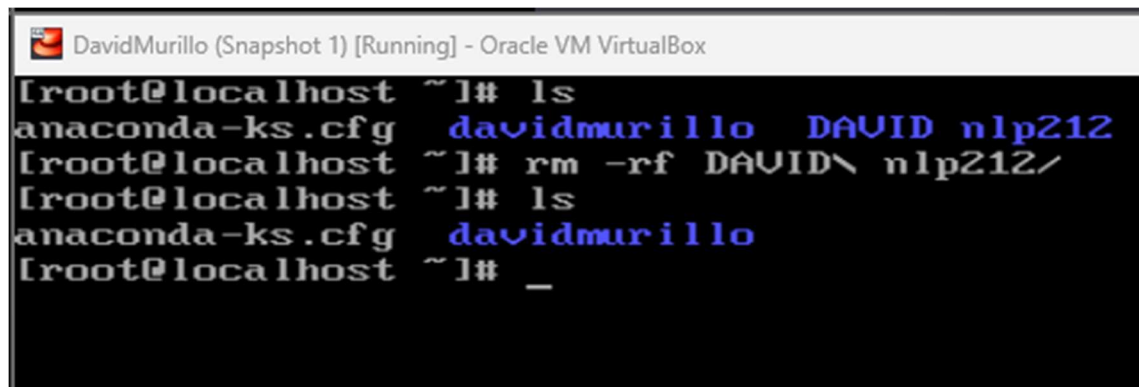
Next, I created a new folder in the home directory, and named it after myself. After which, I copied the previous folder and its contents into the newly created folder.

```
[root@localhost ~]# cp -R DAVID\ nlp212/ davidmurillo/
[root@localhost ~]# ls davidmurillo/
DAVID  nlp212
[root@localhost ~]# ls davidmurillo/DAVID\ nlp212/
davidmurillo
[root@localhost ~]# _
```

*I used the 'cp' command to copy the file and used '-R' to specify that I'm copying a directory, and I specified the file path to ensure that it is copied into the new directory.*

### **Step 4: Remove original folder.**

Next, I removed the original folder.



```
[root@localhost ~]# ls
anaconda-ks.cfg  davidmurillo  DAVID  nlp212
[root@localhost ~]# rm -rf DAVID\ nlp212/
[root@localhost ~]# ls
anaconda-ks.cfg  davidmurillo
[root@localhost ~]# _
```

*I used the 'rm' command with the option '-r' to remove the directory. I also used the 'f' option to confirm the deletion of all files within the directory.*

### **Step 5: Add a sentence to the created file.**

Next, I used the terminal to add a sentence to the file I had previously created using the touch command.

```
[root@localhost ~]# echo "hello world" > davidmurillo/DAVID\ nlp212/davidmurillo
[root@localhost ~]# cat davidmurillo/DAVID\ nlp212/davidmurillo
hello world
[root@localhost ~]#
```

I used the echo command to enter the text I wanted to add into the file, then redirected the output using the '>' into the file path of the file I wanted to edit. To confirm the changes, I used the cat command to confirm the changes.

### Step 6: Change the color of directories.

Next, I used the grep command to search the contents of the /etc/DIR\_COLORS file, in order to find "DIR 01;34" which is the setting for the directory colors on CentOS. I also needed to redirect the line "DIR 01;34" into the file named after me.



```
DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox
[root@localhost ~]# cat /etc/DIR_COLORS | grep "DIR 01;34" >> /root/
anaconda-ks.cfg .bash_logout .bashrc davidmurillo/ .tcshrc
.bash_history .bash_profile .cshrc .ssh/ .viminfo
[root@localhost ~]# cat /etc/DIR_COLORS | grep "DIR 01;34" >> /root/davidmurillo/DAVID\ nlp212/david
murillo
[root@localhost ~]# cat /root/davidmurillo/DAVID\ nlp212/davidmurillo
hello world
DIR 01;34      # directory
[root@localhost ~]#
```

I used the cat command to print the contents of the /etc/DIR\_COLORS directory and then used a pipe '|' to send the output of 'cat' into the 'grep' command. Once on 'grep' I entered the keyword "DIR 01;34" to find every instance of that text in the file. Next, I used the >> redirect the results from the grep command onto the text file I had named after myself. I used '>>' instead of '>' in order to append, instead of overwrite, the contents of the file.

### Step 7: Install vim.

Next, I needed to install vim. I had already installed vim, so for this step I simply updated and upgraded the file editor.

```
DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox
[root@localhost ~]# yum update vim
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirror.lstn.net
 * extras: centos-distro.1gservers.com
 * updates: forsystems.mm.fcix.net
base                                | 3.6 kB  00:00:00
extras                              | 2.9 kB  00:00:00
updates                             | 2.9 kB  00:00:00
updates/7/x86_64/primary_db        | 25 MB  00:00:01
No packages marked for update
[root@localhost ~]# _
```

I used the 'yum update vim' to check if there were any newer versions of vim. If there were, the command would also update vim to the latest available version. Because there were no new versions available there was no point in running the 'yum upgrade' command as essentially, the commands have the same function, except upgrade also removes obsolete files from the previous versions.

### Step 8: Edit directory color settings.

Next, I used the vim text editor to edit the file /etc/DIR\_COLORS file. I needed to search for and edit the 01;34 portion of the file in order to change the color displayed for directories.

```
DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox
[root@localhost ~]#
[root@localhost ~]# vim /etc/DIR_COLORS_
```

First, I entered "vim etc/DIR\_COLORS" to open the file.

```
STICKY_OTHER_WRITABLE  ;42 # dir that is sticky
OTHER_WRITABLE 34;42 # dir that is other-writable
STICKY 37;44      # dir with the sticky bit set (+t

# This is for files with execute permission:
/DIR 01;34_
```

Once the file was open, I entered '/DIR 01;34' in order to keyword search for the location of "01;34" in the file.

```

#FILE 00      # normal file, use no color at all
RESET 0 # reset to "normal" color
DIR 01:34    # directory
LINK 01:36   # symbolic link (If you set this to 'target' instead of a
              # numerical value, the color is as for the file pointed to.)
MULTIHARDLINK 00      # regular file with more than one link

```

Once I found "01:34," I pressed 'i' to begin editing the file.

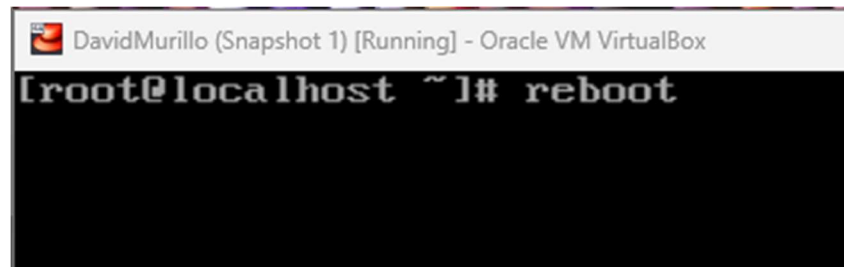
```

RESET 0 # reset to "normal" color
DIR 01:33    # directory
LINK 01:36   # symbolic link (If you set th
              # numerical value, the color i
MULTIHARDLINK 00      # regular file with mo
FIFO 40:33    # pipe
SOCK 01:35    # socket
DOOR 01:35    # door
BLK 40:33:01  # block device driver
CHR 40:33:01  # character device driver
ORPHAN 40:31:01 # symlink to nonexistent file
MISSING 01:05:37:41 # ... and the files they p
SETUID 37:41   # file that is setuid (u+s)
SETGID   :43   # file that is setgid (g+s)
CAPABILITY   :41   # file with capability
STICKY_OTHER_WRITABLE   :42 # dir that is stic
OTHER_WRITABLE 34:42 # dir that is other-writa
STICKY 37:44   # dir with the sticky bit set

# This is for files with execute permission:
:wq!

```

I changed 34 to 33 as those digits control the color of the directory, and I wanted to change the color to yellow, so I replaced 34 with 33. Once I made the changes, I entered ":wq!" to save the changes.



```

DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox
[root@localhost ~]# reboot

```

Although the changes were made, for them to go into effect, the system required rebooting.

```
DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox

CentOS Linux 7 (Core)
Kernel 3.10.0-1160.108.1.el7.x86_64 on an x86_64

localhost login: root
Password:
Last login: Tue Feb  6 21:29:43 on tty1
[root@localhost ~]# ls
anaconda-ks.cfg  davidmurillo
[root@localhost ~]# _
```

Upon reboot and logging back in, I used the 'ls' command to view the directory in my home directory, to verify that the changes were made successfully. The system printed my directory in yellow, revealing that the change was made successfully.

### Step 9: Edit the bootloader.

Next, I once again rebooted the computer, used the arrow keys to prevent the OS from loading, and pressed 'e' to edit the bootloader.

```
DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox

CentOS Linux (3.10.0-1160.108.1.el7.x86_64) 7 (Core)
CentOS Linux (3.10.0-957.12.2.el7.x86_64) 7 (Core)
CentOS Linux (0-rescue-9c00a73d05bf1244a28c29b7f224707f) 7 (Core)
```

I pressed 'e' to edit the first option for bootloader settings.

```
DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox

insmod xfs
set root='hd0,msdos1'
if [ x${feature_platform_search_hint} = xy ]; then
    search --no-floppy --fs-uuid --set=root --hint-bios=hd0,msdos1 --hin\
t-efi=hd0,msdos1 --hint-baremetal=ahci0,msdos1 --hint='hd0,msdos1' ccafb97a-5\
af7-487f-8733-bd0700f849d8
else
    search --no-floppy --fs-uuid --set=root ccafb97a-5af7-487f-8733-bd07\
00f849d8
fi
linux16 /vmlinuz-3.10.0-1160.108.1.el7.x86_64 root=/dev/mapper/centos-\
root ro crashkernel=auto rd.lvm.lv=centos/root rd.lvm.lv=centos/swap rhgb quiet\
LANG=en_US.UTF-8
initrd16 /initramfs-3.10.0-1160.108.1.el7.x86_64.img

Press Ctrl-x to start, Ctrl-c for a command prompt or Escape to
discard edits and return to the menu. Pressing Tab lists
possible completions.
```

Next, I looked for the “root ro” section of the bootloader settings and made changes. The root ro means that, during the booting process, the root filesystem is read only.

```
DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox

insmod xfs
set root='hd0,msdos1'
if [ x${feature_platform_search_hint} = xy ]; then
    search --no-floppy --fs-uuid --set=root --hint-bios=hd0,msdos1 --hin\
t-efi=hd0,msdos1 --hint-baremetal=ahci0,msdos1 --hint='hd0,msdos1' ccafb97a-5\
af7-487f-8733-bd0700f849d8
else
    search --no-floppy --fs-uuid --set=root ccafb97a-5af7-487f-8733-bd07\
00f849d8
fi
linux16 /vmlinuz-3.10.0-1160.108.1.el7.x86_64 root=/dev/mapper/centos-\
root rw init=/sysroot/bin/sh crashkernel=auto rd.lvm.lv=centos/root rd.lvm.lv=\
centos/swap rhgb quiet LANG=en_US.UTF-8
initrd16 /initramfs-3.10.0-1160.108.1.el7.x86_64.img

Press Ctrl-x to start, Ctrl-c for a command prompt or Escape to
discard edits and return to the menu. Pressing Tab lists
possible completions.
```

I made the following changes: I changed ‘ro’ to ‘rw’ to be able to read and write. Next, right beside it, I entered ‘init=/sysroot/bin/sh’. This line tells the kernel to use the initial process to run at boot time, instead of the default system and service manager.

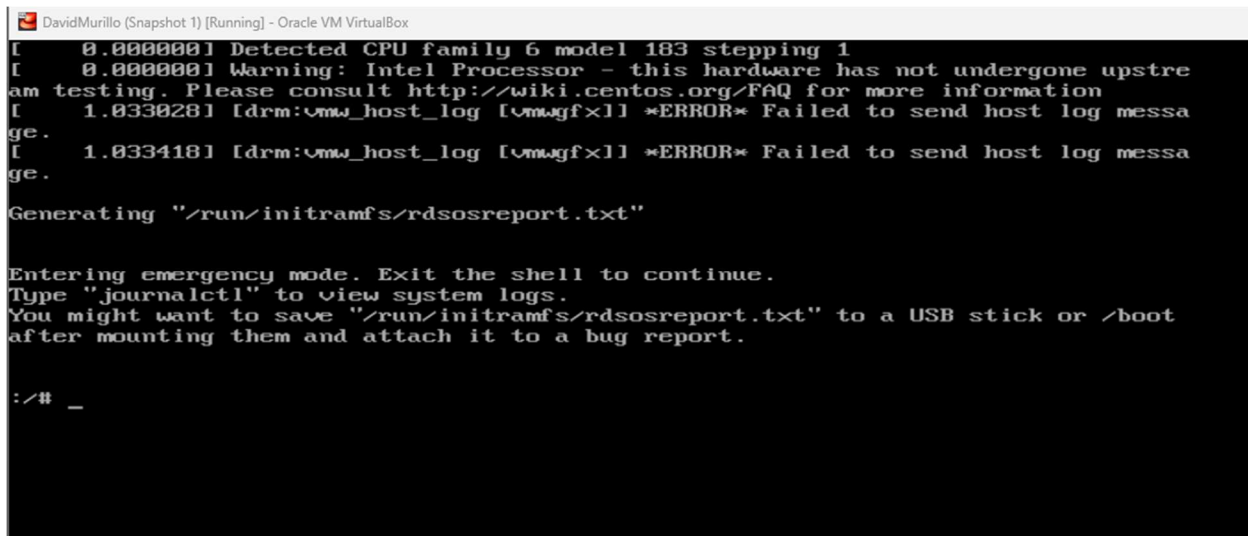
According to Geeks for Geeks, ‘init’ stands for initialization, and is the last step of the kernel boot system. Adding this parameter specifies that what follows is the first program that the kernel should execute.



According to Stack Overflow, 'sysroot' is the root directory during the booting process. Therefore, /sysroot/bin/sh is a file path that leads to a shell program within the root directory. Because I entered 'init' before the /sysroot line, I directed the system to start a shell environment upon booting.

### Step 10: Single User Mode.

Next, I pressed Ctrl + x to enter single user mode. According to "Red Hat Customer Portal", single user mode is a Linux environment which allows the user to recover the system from problems that cannot be resolved within a networked, multi-user environment.



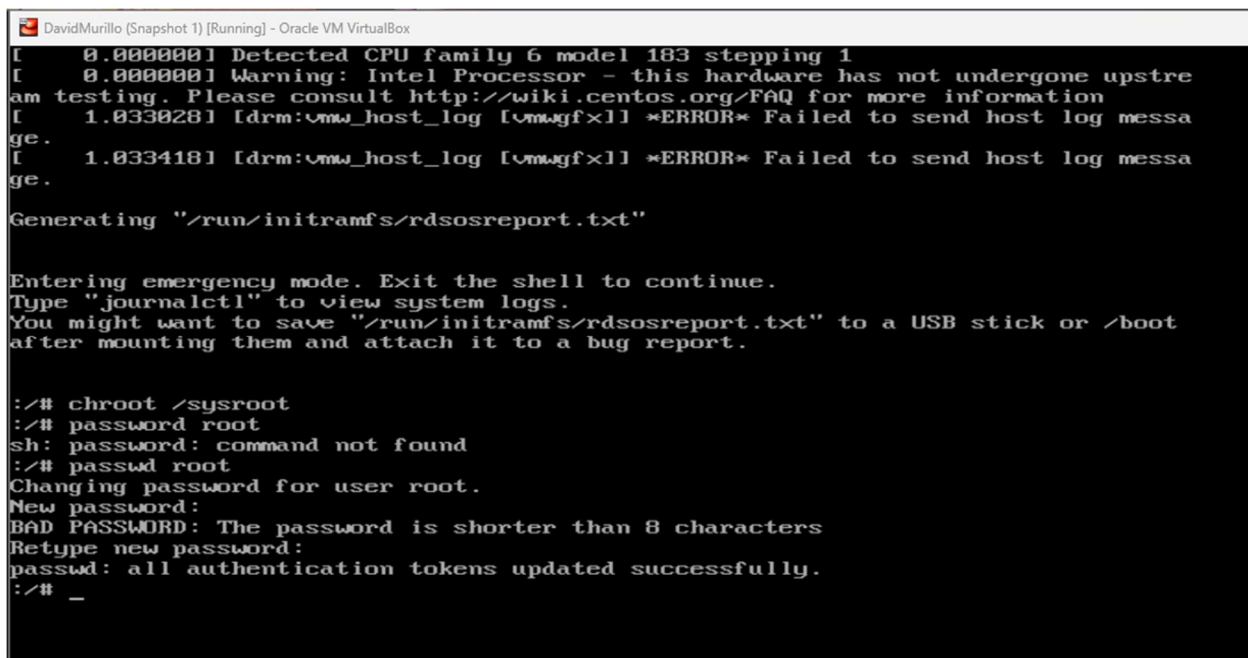
```
DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox
[ 0.000000] Detected CPU family 6 model 183 stepping 1
[ 0.000000] Warning: Intel Processor - this hardware has not undergone upstream testing. Please consult http://wiki.centos.org/FAQ for more information
[ 1.033028] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log message.
[ 1.033418] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log message.

Generating "/run/initramfs/rdsosreport.txt"

Entering emergency mode. Exit the shell to continue.
Type "journalctl" to view system logs.
You might want to save "/run/initramfs/rdsosreport.txt" to a USB stick or /boot after mounting them and attach it to a bug report.

:/# _
```

Screenshot of the 'single user mode' shell environment.



```
DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox
[ 0.000000] Detected CPU family 6 model 183 stepping 1
[ 0.000000] Warning: Intel Processor - this hardware has not undergone upstream testing. Please consult http://wiki.centos.org/FAQ for more information
[ 1.033028] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log message.
[ 1.033418] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log message.

Generating "/run/initramfs/rdsosreport.txt"

Entering emergency mode. Exit the shell to continue.
Type "journalctl" to view system logs.
You might want to save "/run/initramfs/rdsosreport.txt" to a USB stick or /boot after mounting them and attach it to a bug report.

:/# chroot /sysroot
:/# password root
sh: password: command not found
:/# passwd root
Changing password for user root.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: all authentication tokens updated successfully.
:/# _
```

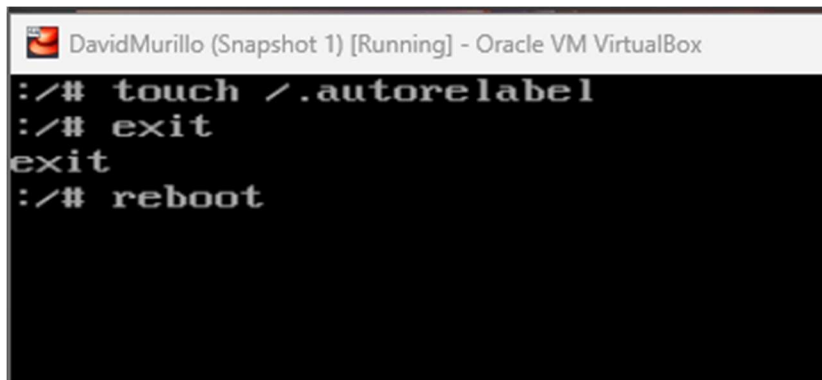
Once in the shell environment, I entered the "chroot" command. After, I attempted to enter the 'passwd root' command but I misspelled it at first, so I retyped it and successfully created a new password for the root user. The 'passwd root' command creates a new password for the root user.



According to Gentoo Wiki, the 'chroot' command is a command that changes the apparent root directory to create a new environment known as a "chroot jail." A user inside a chroot jail cannot see or access files outside of the environment they have been locked into. This is useful for safe system modifications as it reduces the risk of damaging the entire system.

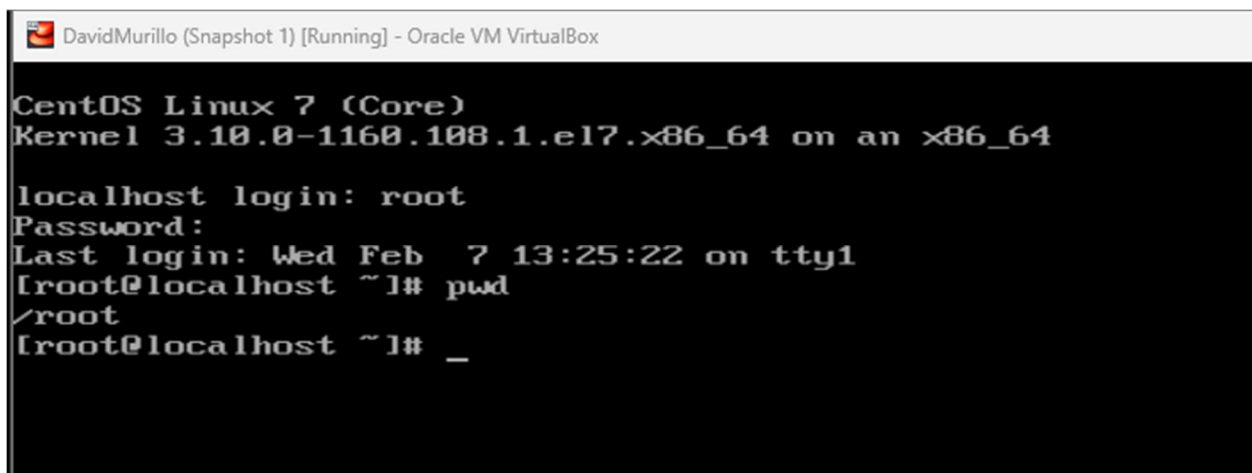
#### **Step 11: Save the change.**

To save the password change, I needed to create an .autorelabel file which will automatically relabel the filesystem for SELinux.

A terminal window titled "DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox" showing a chroot session. The user enters the following commands: touch /.autorelabel, exit, and reboot. The prompt is always :/#.

```
:/# touch /.autorelabel
:/# exit
exit
:/# reboot
```

*I used the 'touch' command to create the '.autorelabel' file, then used exit to exit out of the chroot jail, and rebooted the computer so that the changes could go into effect.*

A terminal window titled "DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox" showing the CentOS Linux 7 boot process. It displays the kernel version, login prompt for root, password prompt, last login time, and the output of the pwd command showing the root directory.

```
CentOS Linux 7 (Core)
Kernel 3.10.0-1160.108.1.el7.x86_64 on an x86_64

localhost login: root
Password:
Last login: Wed Feb  7 13:25:22 on tty1
[root@localhost ~]# pwd
/root
[root@localhost ~]# _
```

*The booting process took a little bit longer than usual as the file system was being relabeled, but once the OS was loaded, I was able successfully log in using the new password.*

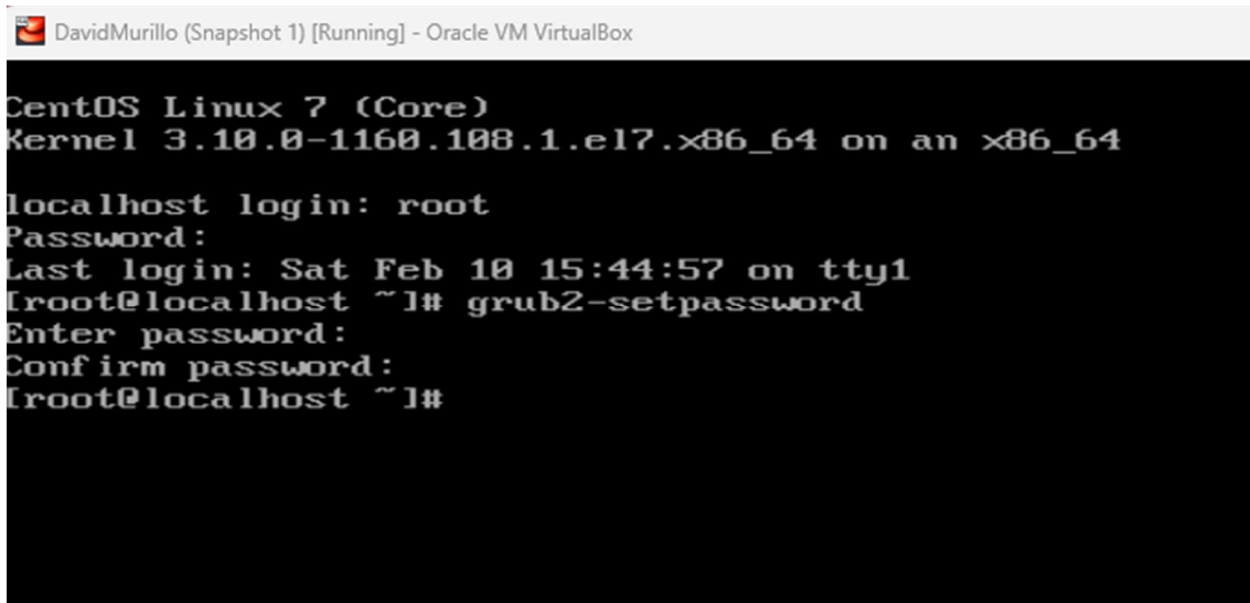
#### **Question 1: Why can a user who can access the bootloader bypass the password authentication of an Operating System?**

When booting a computer, the booting process goes as follows: First the BIOS initializes hardware and then locates and runs the bootloader. The bootloader's responsibility is to load the OS onto memory. Because the bootloader operates at a level before the Operating System is fully loaded, the bootloader is responsible for initiating the OS startup process and any pre-boot configurations. Therefore, any

security features that the Operating System may have will not defend against system changes made from the bootloader.

### **Step 12: Generate a password for GRUB.**

To enhance the security of my machine, I took measures to prevent unauthorized access, starting with the hardening of the bootloader. This involved implementing a GRUB password, as GRUB serves as the bootloader and boot-manager for modern Linux distributions.

A terminal window titled "DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox" showing the CentOS Linux 7 login process. The user logs in as root and runs the grub2-setpassword command to set a password for the bootloader. The terminal output is as follows:

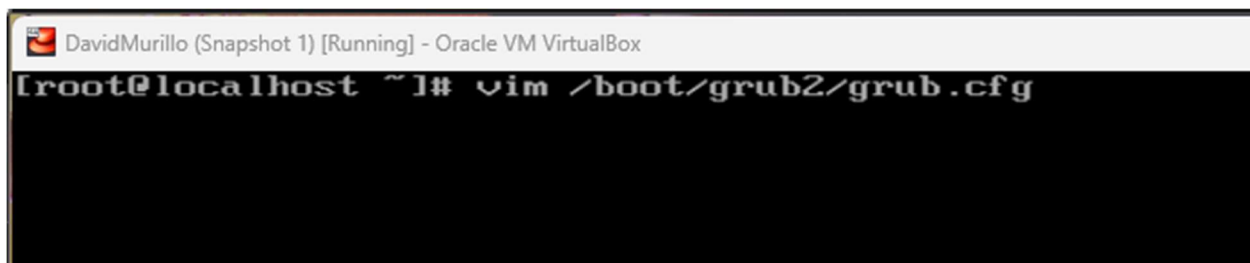
```
CentOS Linux 7 (Core)
Kernel 3.10.0-1160.108.1.el7.x86_64 on an x86_64

localhost login: root
Password:
Last login: Sat Feb 10 15:44:57 on tty1
[root@localhost ~]# grub2-setpassword
Enter password:
Confirm password:
[root@localhost ~]#
```

*First, I executed the grub2-setpassword command and then created a strong password for the bootloader.*

### **Step 13: Enable the password for the Boot entry.**

Next, I opened the “/boot/grub2/grub.cfg” file with the vim text editor.

A terminal window titled "DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox" showing the user running the vim command to open the /boot/grub2/grub.cfg file. The terminal output is as follows:

```
[root@localhost ~]# vim /boot/grub2/grub.cfg
```

*I used the vim command and used the absolute reference to the “grub.cfg” file to open it with the vim text editor.*

```
DavidMunillo (Snapshot 1) [Running] - Oracle VM VirtualBox
menuentry 'CentOS Linux (3.10.0-1160.108.1.el7.x86_64) ? (Core)' --class centos --class gnu-linux --
class gnu --class os --unrestricted $menuentry_id_option 'gnulinux-3.10.0-957.12.2.el7.x86_64-advanced-e0009eef-a04b-49ab-b229-5aa8cac092d2' {
    load_video
    set gfxpayload=keep
    insmod gzio
    insmod part_msdos
    insmod xfs
    set root='hd0,msdos1'
    if [ x$feature_platform_search_hint = xy ]; then
        search --no-floppy --fs-uuid --set=root --hint-bios=hd0,msdos1 --hint-efi=hd0,msdos1 --hin
t-baremetal=ahci0,msdos1 --hint='hd0,msdos1' ccafb97a-5af7-487f-8733-bd0700f849d8
    else
        search --no-floppy --fs-uuid --set=root ccafb97a-5af7-487f-8733-bd0700f849d8
    fi
    linux16 /vmlinuz-3.10.0-1160.108.1.el7.x86_64 root=/dev/mapper/centos-root ro crashkernel=au
to rd.lvm.lv=centos/root rd.lvm.lv=centos/swap rhgb quiet LANG=en_US.UTF-8
    initrd16 /initramfs-3.10.0-1160.108.1.el7.x86_64.img
}
menuentry 'CentOS Linux (3.10.0-957.12.2.el7.x86_64) ? (Core)' --class centos --class gnu-linux --cl
ass gnu --class os --unrestricted $menuentry_id_option 'gnulinux-3.10.0-957.12.2.el7.x86_64-advanced-e0009eef-a04b-49ab-b229-5aa8cac092d2' {
    load_video
    set gfxpayload=keep
    insmod gzio
    insmod part_msdos
    insmod xfs
    set root='hd0,msdos1'
    if [ x$feature_platform_search_hint = xy ]; then
        search --no-floppy --fs-uuid --set=root --hint-bios=hd0,msdos1 --hint-efi=hd0,msdos1 --hin
t-baremetal=ahci0,msdos1 --hint='hd0,msdos1' ccafb97a-5af7-487f-8733-bd0700f849d8
    else
        search --no-floppy --fs-uuid --set=root ccafb97a-5af7-487f-8733-bd0700f849d8
    fi
    linux16 /vmlinuz-3.10.0-957.12.2.el7.x86_64 root=/dev/mapper/centos-root ro crashkernel=auto
rd.lvm.lv=centos/root rd.lvm.lv=centos/swap rhgb quiet LANG=en_US.UTF-8
}
103,136 67%
```

Once on the vim editor, I navigated the file using the keyword “menuentry.”

By using “menuentry” as the keyword, I was able to quickly find the 2 boot entries. Both had “--unrestricted” flags, meaning that any could enter the boot entries without having to enter a password. Therefore, I removed the flag to one of the entries, to require all users to enter a password.

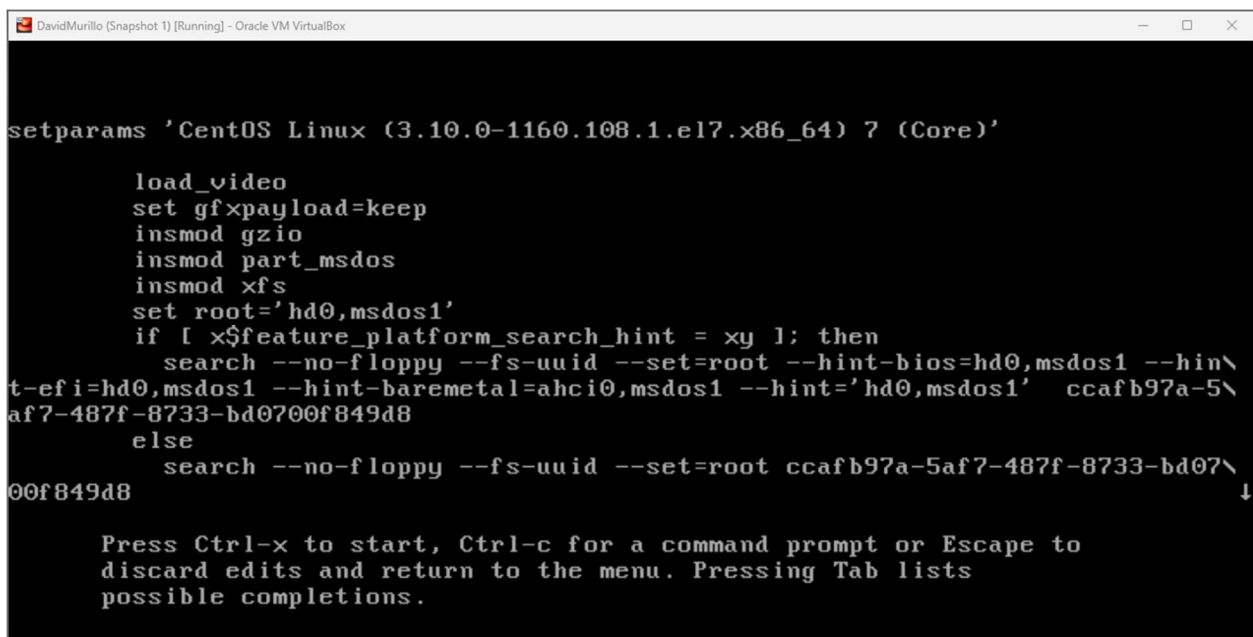
```
### BEGIN /etc/grub.d/10_linux ###
menuentry 'CentOS Linux (3.10.0-1160.108.1.el7.x86_64) ? (Core)' --class centos --class gnu-linux --
class gnu --class os $menuentry_id_option 'gnulinux-3.10.0-957.12.2.el7.x86_64-advanced-e0009eef-a04
b-49ab-b229-5aa8cac092d2' {
    load_video
    set gfxpayload=keep
    insmod gzio
    insmod part_msdos
    insmod xfs
    set root='hd0,msdos1'
    if [ x$feature_platform_search_hint = xy ]; then
        search --no-floppy --fs-uuid --set=root --hint-bios=hd0,msdos1 --hint-efi=hd0,msdos1 --hin
```

I removed the unrestricted flag from the file and saved the change.

Next, I rebooted the system and opened the bootloader settings for the boot entry who's flag I removed.

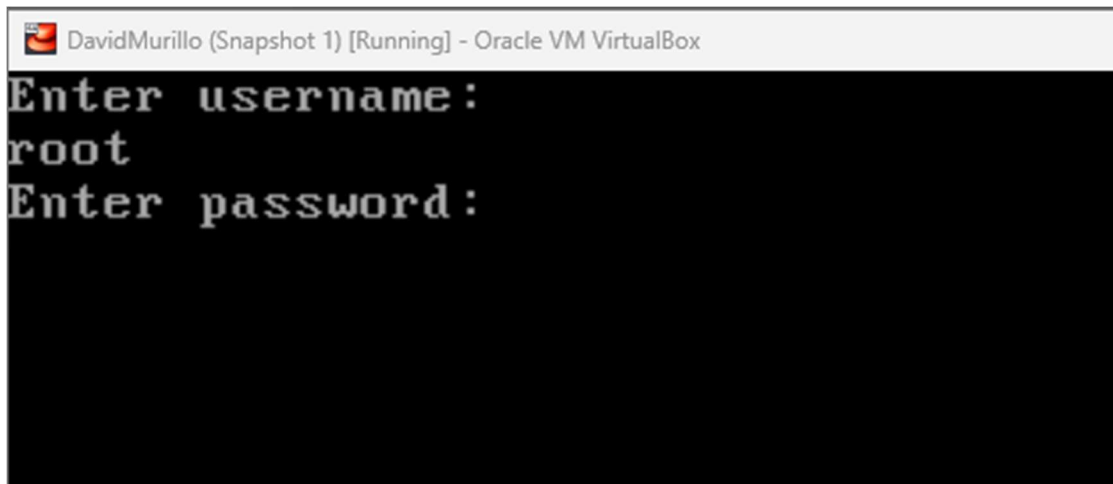


Once I pressed "e" to edit the boot entry, I was required to give my username, and the password which I had previously created.



Upon entering my username and password, I was able to successfully access the bootloader settings.

Next, I rebooted the system and tried to access the OS login screen.



*Prior to accessing the login screen, I was prompted to enter my user and the GRUB password I had created.*

**Question 2: What is the distinction between setting a GRUB password and removing the "--unrestricted" flag in terms of system protection, and what specific security does each action provide?**

Creating a GRUB password only protects against unauthorized editing of the bootloader settings. By leaving the "--unrestricted" flag, users can still load the bootloader without needing to enter a password. Therefore, removing the flag enhances bootloader security, requiring a password for both booting and editing.

**Question 3: According to the computer's booting process, if the bootloader is hardened with a password, does it mean that the computer/server is safe?**

Even with the improved security of a bootloader password, the computer is still vulnerable to lower-level attacks. The reason I created a bootloader password in the first place is that, despite the security features of an operating system, the OS remains vulnerable as those features are not yet loaded when on the bootloader. The bootloader is also susceptible to lower-level attacks from the BIOS. Because the booting process begins with the BIOS, it is important to harden it, in order to protect the rest of the system.

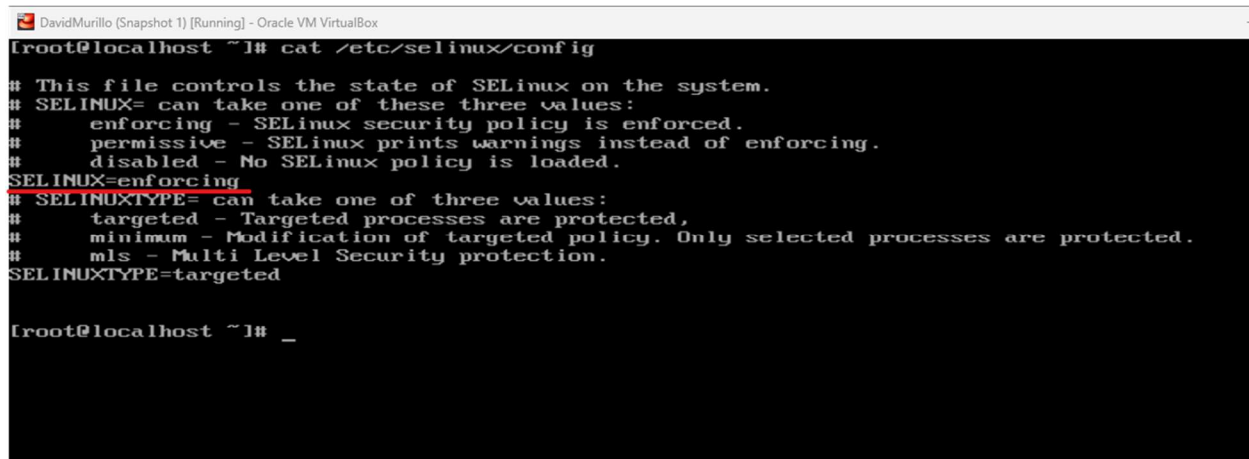
**Question 4: We assume that a server runs dual boot (i.e., two Operating Systems, such as CentOS 6 and CentOS 8, are installed in the same computer, and the computer can be booted via one of the Operating Systems each time). Do you think a root user of CentOS 6 is able to change CentOS 8's root password? If the root user of CentOS 6 sets a GRUB password and enables this password (by removing the "--unrestricted" flag) to all the boot entries, is the root user in CentOS 8 still able to log on CentOS 8 (assume BIOS is not accessible)?**

In the scenario where the dual boot system were using GRUB to run CentOS 6 and CentOS 8 on the same machine, the actions of users on one OS may affect the users of the counterpart OS. One way the users of the other operating system may be affected is, for example, if the root user of CentOS 6 creates a bootloader password and removes the flag for both entries. This action results in a global password required to be entered by both users of CentOS 6 and 8. If the users of CentOS 8 are not informed of the new password, they will be unable to enter the OS nor the bootloader settings. Also, because the CentOS 6 user holds the global bootloader password, they have full access to both operating system's

bootloader settings. Therefore, the user may enter the CentOS 8 bootloader settings and follow the same steps I followed during this lab, and change the root password. Thus, if a malicious actor were to have full access to one OS, it may compromise the security of the other operating system on the boot manager.

### Step 13: Enable SELinux

In this step, I needed to enable SELinux. According to “Red Hat”, SELinux is a security framework for Linux systems that enhances access control mechanisms, allowing administrators to define and manage policies that govern how users and processes can access system resources.

A terminal window titled "DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox" shows a root user at localhost. The user enters the command "cat /etc/selinux/config". The output displays the SELinux configuration file content, which includes a comment about the file's purpose, a list of three possible SELinux states (enforcing, permissive, disabled), and the current state "SELINUX=enforcing". It also lists three possible SELinux types (targeted, minimum, mls) and the current type "SELINUXTYPE=targeted".

```
[root@localhost ~]# cat /etc/selinux/config
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#   enforcing - SELinux security policy is enforced.
#   permissive - SELinux prints warnings instead of enforcing.
#   disabled - No SELinux policy is loaded.
SELINUX=enforcing
# SELINUXTYPE= can take one of three values:
#   targeted - Targeted processes are protected,
#   minimum - Modification of targeted policy. Only selected processes are protected.
#   mls - Multi Level Security protection.
SELINUXTYPE=targeted

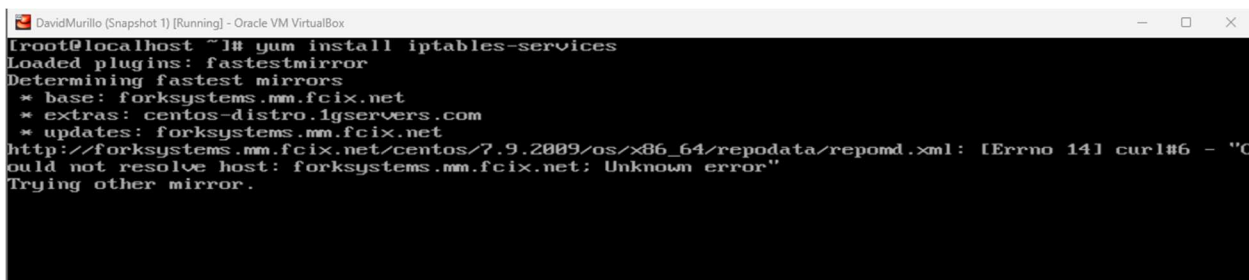
[root@localhost ~]# _
```

To begin, I used the cat command to cat the SELinux config file in order to determine whether SELinux was on or not.

The cat command printed out “SELINUX=enforcing” meaning that SELinux was already actively enforcing on my machine.

### Step 14: Install Iptables.

In this step, I needed to install and enable iptables to replace the default firewall interface. After which, I needed to turn off iptables and reestablish firewalld.

A terminal window titled "DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox" shows a root user at localhost. The user enters the command "yum install iptables-services". The output shows the yum process loading plugins, determining fastest mirrors, and listing mirrors. It then shows an error message: "http://forksystems.mm.fcix.net/centos/7.9.2009/os/x86\_64/repodata/repomd.xml: [Errno 14] curl#6 - 'Could not resolve host: forksystems.mm.fcix.net; Unknown error'", followed by "Trying other mirror.".

```
[root@localhost ~]# yum install iptables-services
Loaded plugins: fastestmirror
Determining fastest mirrors
 * base: forksystems.mm.fcix.net
 * extras: centos-dist.0.1gserver.com
 * updates: forksystems.mm.fcix.net
http://forksystems.mm.fcix.net/centos/7.9.2009/os/x86_64/repodata/repomd.xml: [Errno 14] curl#6 - "
ould not resolve host: forksystems.mm.fcix.net; Unknown error"
Trying other mirror.
```

To install iptables, I used yum install and entered “yum install iptables-services.”



```
DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox
* base: forksystems.mm.fcix.net
* extras: centos-distro.igservers.com
* updates: forksystems.mm.fcix.net
Resolving Dependencies
--> Running transaction check
---> Package iptables-services.x86_64 0:1.4.21-35.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

=====
Package                               Arch          Version           Repository        Size
=====
Installing:
iptables-services                     x86_64        1.4.21-35.el7     base              52 k
=====

Transaction Summary
=====
Install 1 Package

Total download size: 52 k
Installed size: 23 k
Is this ok [y/d/N]: yes
Downloading packages:
iptables-services-1.4.21-35.el7.x86_64.rpm                                | 52 kB  00:00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : iptables-services-1.4.21-35.el7.x86_64                    1/1
  Verifying  : iptables-services-1.4.21-35.el7.x86_64                    1/1

Installed:
iptables-services.x86_64 0:1.4.21-35.el7

Complete!
[root@localhost ~]#
```

*It took around 15 minutes but as shown at the bottom of the screenshot, iptables was successfully installed.*

### Step 15: Disable Firewall.

Next, I needed to disable firewall.

```
DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox
[root@localhost ~]# systemctl disable firewalld
[root@localhost ~]# systemctl stop firewalld
[root@localhost ~]#
```

*To disable firewalld, I used the systemctl command to both disable and stop the firewalld service.*

### Question 5: What is the difference between systemctl disable and systemctl stop?

Using the “man” command, you can view the manual for a specified command. According to the CentOS manual, systemctl disable prevents a service from starting automatically during system boot but does not immediately stop the service if it is running. On the other hand, systemctl stop immediately halts a running service but does not prevent it from running again upon next boot.

### Step 16: Enable and Start Iptables.

Next, I enabled and started iptables by using systemctl commands.

```
DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox
[root@localhost ~]# systemctl enable iptables
Created symlink from /etc/systemd/system/basic.target.wants/iptables.service to /usr/lib/systemd/system/iptables.service.
[root@localhost ~]# systemctl start iptables
[root@localhost ~]#
```

*I entered “systemctl enable iptables” and “systemctl start iptables” to enable and start iptables.*

There were no errors with the command, but I wanted to verify that iptables was active. To confirm that iptables was running, I used the ‘status’ command.

```
DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox
[root@localhost ~]# systemctl status iptables
■ iptables.service - IPv4 firewall with iptables
   Loaded: loaded (/usr/lib/systemd/system/iptables.service; enabled; vendor preset: disabled)
   Active: active (exited) since Sun 2024-02-11 12:56:07 CST; 4min 58s ago
     Process: 1502 ExecStart=/usr/libexec/iptables/iptables.init start (code=exited, status=0/SUCCESS)
    Main PID: 1502 (code=exited, status=0/SUCCESS)

Feb 11 12:56:07 localhost.localdomain systemd[1]: Starting IPv4 firewall with iptables...
Feb 11 12:56:07 localhost.localdomain iptables.init[1502]: iptables: Applying firewall rules: [... ]
Feb 11 12:56:07 localhost.localdomain systemd[1]: Started IPv4 firewall with iptables.
Hint: Some lines were ellipsized, use -l to show in full.
[root@localhost ~]# _
```

*The output indicated it was in an active state, thus confirming it had been enabled successfully.*

After confirming that iptables was active, I wanted to view the firewall rules. To view the firewall rules I entered “iptables -L”.

```
DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox
[root@localhost ~]# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source                destination          state
ACCEPT     all  --  anywhere               anywhere             state RELATED,ESTABLISHED
ACCEPT     icmp --  anywhere               anywhere
ACCEPT     all  --  anywhere               anywhere
ACCEPT     tcp  --  anywhere               anywhere             state NEW tcp dpt:ssh
REJECT     all  --  anywhere               anywhere             reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
target     prot opt source                destination          reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
target     prot opt source                destination
[root@localhost ~]# _
```

*These were the firewall rules.*

The firewall is configured to allow ICMP traffic and SSH connections, while it blocks all other incoming and forwarded traffic that does not meet these criteria.

### **Step 17: Turn off Iptables and Reactivate Firewall.**

In this step, I needed to turn off iptables and reestablish the original firewall configuration “firewalld”.

```
DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox
[root@localhost ~]# systemctl stop iptables
[root@localhost ~]# systemctl disable iptables
Removed symlink /etc/systemd/system/basic.target.wants/iptables.service.
```

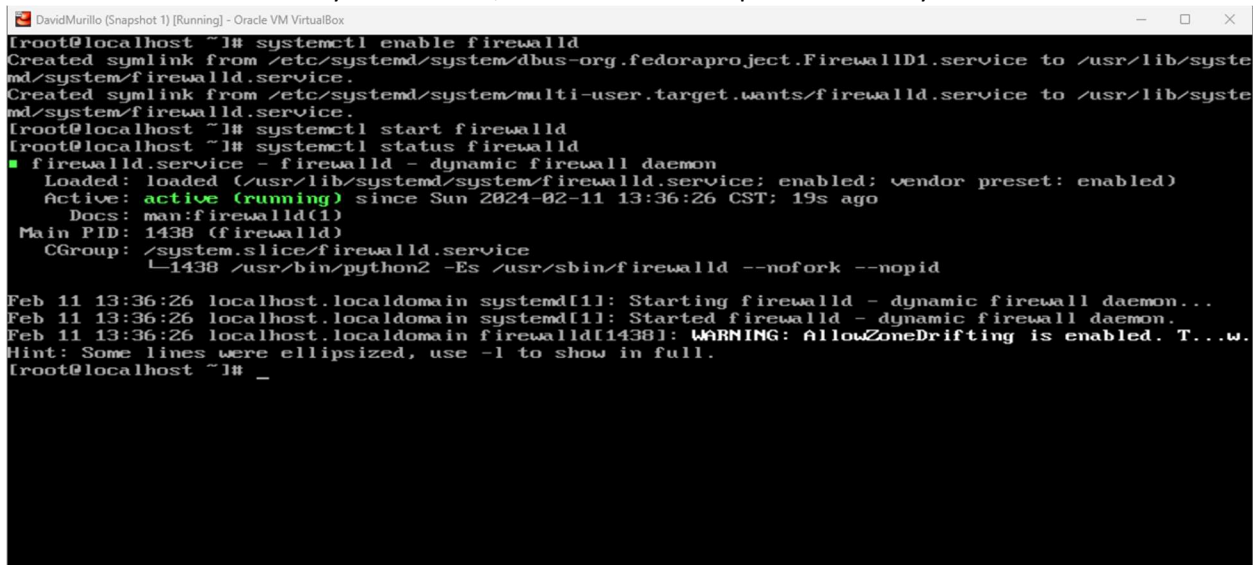
*I used the both the systemctl stop and disable commands to stop iptables from running and preventing it from running upon reboot.*

```
DavidMurillo (Snapshot 1) [Running] - Oracle VM VirtualBox
[root@localhost ~]# systemctl status iptables
■ iptables.service - IPv4 firewall with iptables
   Loaded: loaded (/usr/lib/systemd/systemd/iptables.service; disabled; vendor preset: disabled)
   Active: inactive (dead) since Sun 2024-02-11 13:23:52 CST; 2min 35s ago
   Main PID: 1502 (code=exited, status=0/SUCCESS)

Feb 11 12:56:07 localhost.localdomain systemd[1]: Starting IPv4 firewall with iptables...
Feb 11 12:56:07 localhost.localdomain iptables.init[1502]: iptables: Applying firewall rules: [... ]
Feb 11 12:56:07 localhost.localdomain systemd[1]: Started IPv4 firewall with iptables.
Feb 11 13:23:52 localhost.localdomain systemd[1]: Stopping IPv4 firewall with iptables...
Feb 11 13:23:52 localhost.localdomain iptables.init[1559]: iptables: Setting chains to policy A... ]
Feb 11 13:23:52 localhost.localdomain iptables.init[1559]: iptables: Flushing firewall rules: [... ]
Feb 11 13:23:52 localhost.localdomain systemd[1]: Stopped IPv4 firewall with iptables.
Hint: Some lines were ellipsized, use -l to show in full.
[root@localhost ~]#
```

*I also used the systemctl status command to verify that the service was disabled. The output displayed “inactive (dead)”, indicating that the service was successfully disabled.*

I initiated the firewalld service to start on boot by using the systemctl enable command. Then, I activated the service with systemctl start, and confirmed its operation with systemctl status.

A screenshot of a terminal window titled "DavidMuriillo (Snapshot 1) [Running] - Oracle VM VirtualBox". The terminal shows the following commands and output:

```
[root@localhost ~]# systemctl enable firewalld
Created symlink from /etc/systemd/system/dbus-org.fedoraproject.FirewallD1.service to /usr/lib/systemd/system/firewalld.service.
Created symlink from /etc/systemd/system/multi-user.target.wants/firewalld.service to /usr/lib/systemd/system/firewalld.service.
[root@localhost ~]# systemctl start firewalld
[root@localhost ~]# systemctl status firewalld
■ firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2024-02-11 13:36:26 CST; 19s ago
     Docs: man:firewalld(1)
    Main PID: 1438 (firewalld)
    CGroup: /system.slice/firewalld.service
            └─1438 /usr/bin/python2 -Es /usr/sbin/firewalld --nofork --nopid

Feb 11 13:36:26 localhost.localdomain systemd[1]: Starting firewalld - dynamic firewall daemon...
Feb 11 13:36:26 localhost.localdomain systemd[1]: Started firewalld - dynamic firewall daemon.
Feb 11 13:36:26 localhost.localdomain firewalld[1438]: WARNING: AllowZoneDrifting is enabled. T...w.
Hint: Some lines were ellipsized, use -l to show in full.
[root@localhost ~]# _
```

Systemctl status returned "active (running)", indicating that the service was successfully activated and would remain active upon all boots.

---

## LIMITATIONS/CONCLUSION

In this lab, I successfully implemented several fundamental security measures on a CentOS system, which included configuring Linux commands, securing the boot process through password management, reinforcing the GRUB bootloader's security, enabling and managing SELinux, and administering firewall functionalities with iptables and firewalld. These steps have strengthened the system's defenses against unauthorized access and potential security threats.

---

## REFERENCES

"Init Command in Linux with Examples." GeeksforGeeks, <https://www.geeksforgeeks.org/init-command-in-linux-with-examples/>.

I used this website to understand what init is, what it does, and how I could use it.

"What Is a Sysroot Exactly and How Do I Create One?" Stack Overflow, <https://stackoverflow.com/questions/39920712/what-is-a-sysroot-exactly-and-how-do-i-create-one>.

I used this website to learn about sysroot and what it does.

"Chroot." Gentoo

Wiki, [https://wiki.gentoo.org/wiki/Chroot#:~:text=chroot%20\(Change%20root\)%20is%20a,the%20main%20system's%20root%20directory](https://wiki.gentoo.org/wiki/Chroot#:~:text=chroot%20(Change%20root)%20is%20a,the%20main%20system's%20root%20directory).

I used this website to understand the chroot command and its practicality.

"What is SELinux?" Red Hat, 30 Aug. 2019, <https://www.redhat.com/en/topics/linux/what-is-selinux>.

I used this website to understand what SELinux is and how it works.