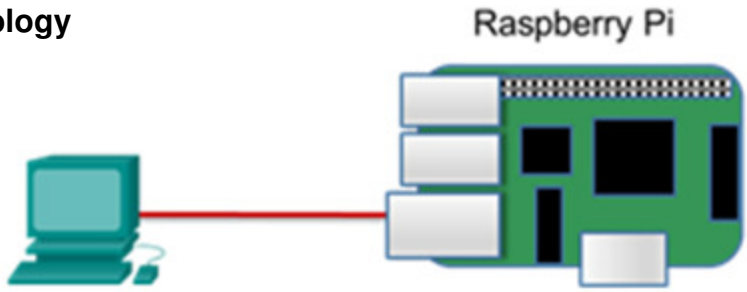
Harden a Raspberry Pi

Alternate to Cisco IoT:Security Lab 1.2.3.3

Topology



Objectives

Part 1: Securing Remote Access

Part 2: Removing the Default Pi User Account

Part 3: Configuring the Uncomplicated Firewall (UFW)

Background/Scenario

The Raspberry Pi is a doorway to the Internet of Things (IoT). In this lab you will take a Raspberry Pi that is acting as an IoT gateway device and perform device hardening. You will harden the Raspberry Pi by following recommended security practices for the Pi OS. You will also limit the network protocols and services allowed to connect to the IoT gateway by activating and configuring the **iptables** firewall
Finally, you will utilize a separate Kali VM with the Kali Linux OS, acting as a threat actor, to test the security of the IoT gateway.

Required Resources

- Raspberry Pi 3 B+ or Pi 4
- 8GB or larger MicroSD card
- Host computer with at least 4 GB of RAM and 15 GB of free disk space
- Oracle VirtualBox (you could use VMware Player instead … instructions not included)
- Kali IoT:Security virtual machine
- Metasploitable virtual machine
- Internet connection
- Ethernet patch cables

Part 1: Securing Remote Access

The Raspberry Pi comes with a default user called “pi” with the default password of “raspberry” which is well known. While this makes it easy to use the system, it is not very secure. Anyone with network access to your Pi could login with these widely known credentials. Furthermore, because the SSH server and HTTP server on the Pi are enabled, unknown users on the network could attempt a connection using these default credentials. While basic security would advise the changing of the password for the “pi” user, having a default username alone is a security risk.

Step 1: Access the Raspberry Pi remotely.

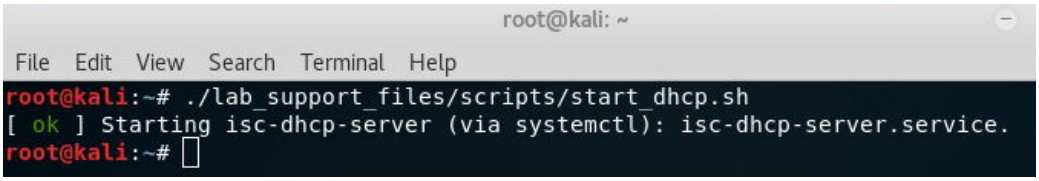
If you have not completed Alt Lab 1.2.3.2, go back and complete the lab before beginning this lab.

a. Start the **Kali VM** and login with the username **root** and the password **toor**.

b. Click on **Terminal** on the left side of the screen.

c. On the **Kali VM**, run the shell script to configure IP addressing. To run the script, at the terminal prompt type the following:

```
root@kali:~# ./lab_support_files/scripts/start_dhcp.sh
```



a. Plug the Pi in to start it up (wait for 3 minutes for the Pi to power up)

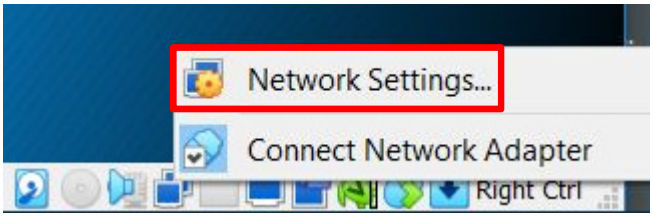
b. Enter the command **fping –A –d –a –q –g 203.0.113.0/24** in a terminal on the Kali VM to determine the IP address of your Raspberry Pi.

Record the IP address of your Raspberry Pi. _____

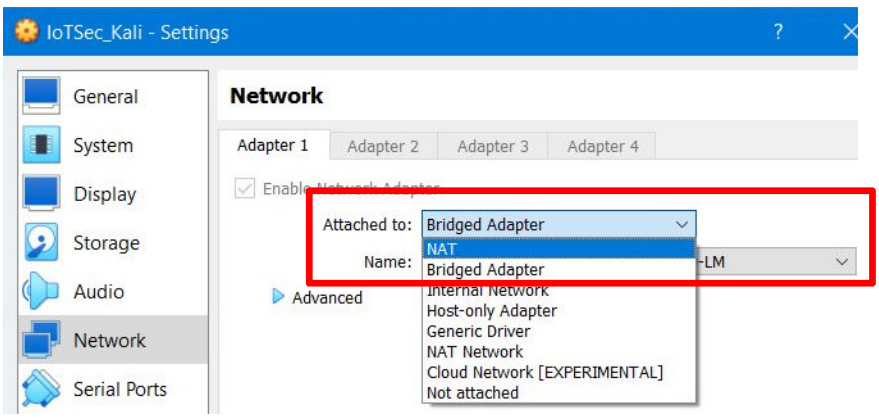
*There is a small bug in VirtualBox's Bridged networking. If you do not see the IP address of the Pi in the **fping** output, do the following:*

- *In the lower right of the VirtualBox window, right-click on the **network adapter icon***

- *Click on **Network Settings**.*



- *In the Network settings screen, click on list arrow in the **Attached to**: box and select **NAT** from the list. Click **OK***



- *Repeat the two previous steps and change the **Attached to**: from **NAT** to **Bridged Adapter**. Click **OK***

*Try the **fping** command again.*

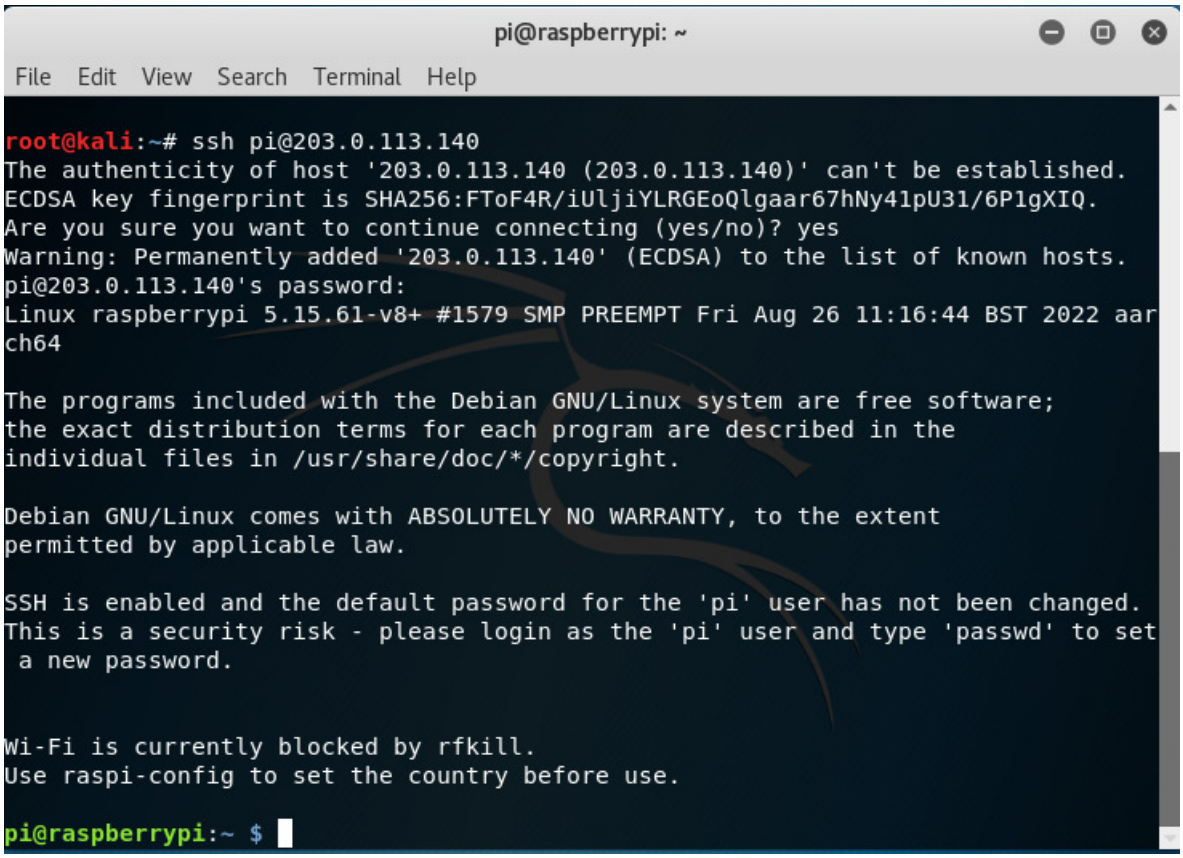
c. The Raspberry Pi has the default username **pi** and password **raspberry**. Use **SSH** to remotely access the Raspberry Pi. The IP address used in this lab is only used as an example, the IP address for your Raspberry Pi may be different. The IP address for your Raspberry Pi will be in the 203.0.113.0/24 subnet

   In the terminal on the **Kali VM** , type the following command to SSH into the Raspberry Pi using the **pi** account.

   root@kali:~# ssh pi@<<Pi's IP address>>

d. When **warned about the authenticity of the host cannot be established**, type **yes**

e. Type **raspberry** for the password



Step 2: Securing the user accounts

As we know, while basic security would advise the changing of the password for the **pi** user, just having a default username is a security risk. Instead, you will create a new user with sudo permissions. After the new user has the appropriate permissions, the default **pi** user can be deleted.

Note: The name of the Raspberry Pi that appears in the terminal will differ depending on the device name that was configured in PL-App launcher when the SD card was made.

a. Add a new user to the Pi with the command **sudo adduser kingbob** in the terminal. Choose a password and press **Enter** on all of the other information fields.

```
pi@raspberrypi:~ $ sudo adduser kingbob
```

b. Give the kingbob user sudo permissions by adding it to the sudo group with the command **sudo adduser kingbob sudo**.

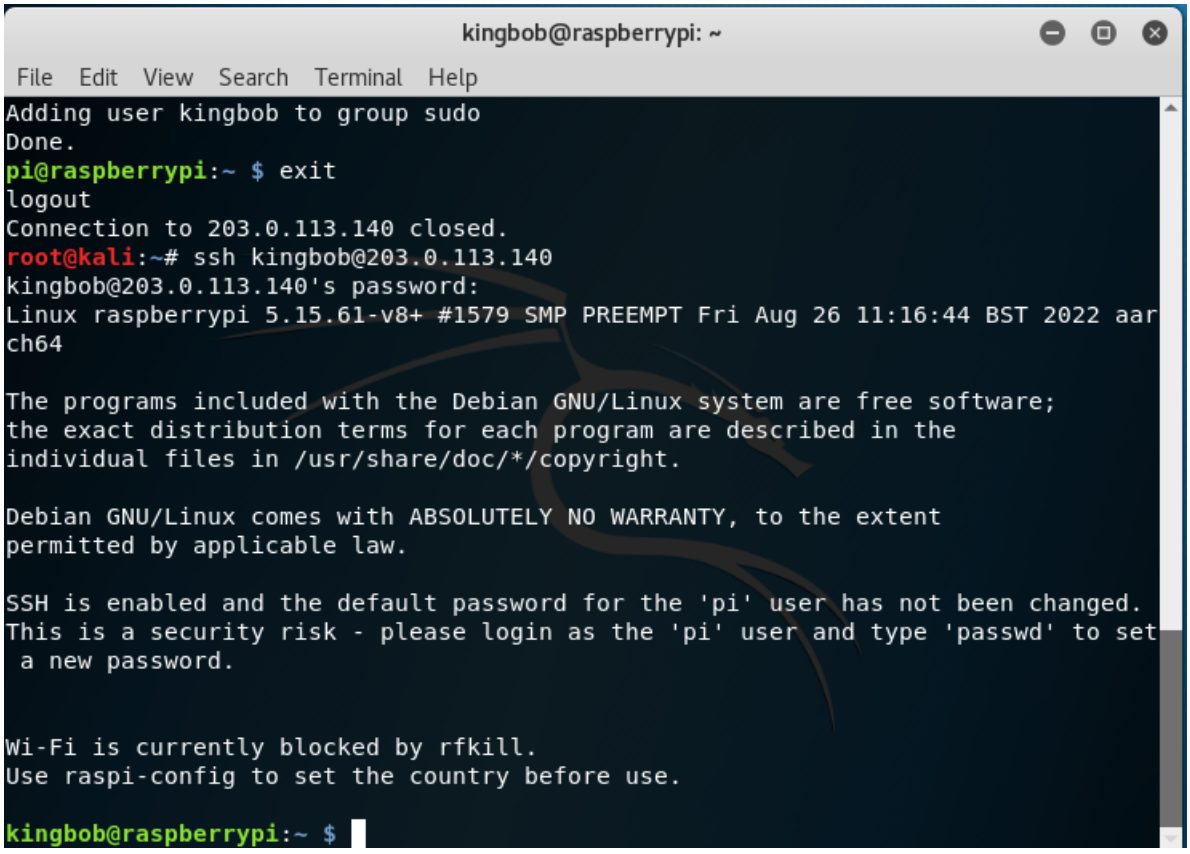
```
pi@raspberrypi:~ $ sudo adduser kingbob sudo
```

c. End the **SSH** session by typing **exit**.

d. **SSH** into the Raspberry Pi using the **kingbob** account, which is permitted to access the Pi via **SSH**.

```
root@kali:~# ssh kingbob@<<Pi's IP address>>
```

e. Take a screenshot of the terminal window at this point



Step 3: Secure remote access.

The implementation of **SSH** as a method for remote access in itself does not provide strong security, the default installation of **SSH** uses a single password, commonly with a default value. To implement stronger security when deploying the SSH service, a **username** and **password** combination should be implemented.

This next step includes the activation of the **SSH** service and restriction of authentication attempts.

a. You will create **kevin**, a standard user, with the command following command. Press **Enter** on all of the information fields

```
kingbob@raspberrypi:~ $ sudo adduser kevin

We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:
    #1) Respect the privacy of others.
    #2) Think before you type.
    #3) With great power comes great responsibility.
[sudo] password for kingbob:
```

b. Now edit the **sshd_config** file located in the /**etc**/**ssh** directory to limit the users that are allowed to access this device using **SSH** by typing the following command: (**nano** is a simple Linux text editor)

```
kingbob@raspberrypi:~ $ sudo nano /etc/ssh/sshd_config
```

c. Add the following two lines to the end of the file (case sensitive):

```
AllowUsers kingbob
DenyUsers kevin
```

Press **Ctrl-O** then press **ENTER** to write the file and press **Ctrl-X** to exit **nano**.

d. To force the **SSH** settings to take effect now, restart the SSH service using **sudo systemctl restart ssh**.

```
kingbob@raspberrypi:~ $ sudo systemctl restart ssh
```

e. End the **SSH** session by typing **exit**.

f. Verify that the **kevin** user account cannot be exploited by a threat actor via the SSH service. Type the following command to attempt an **SSH** connection with the **kevin** username.

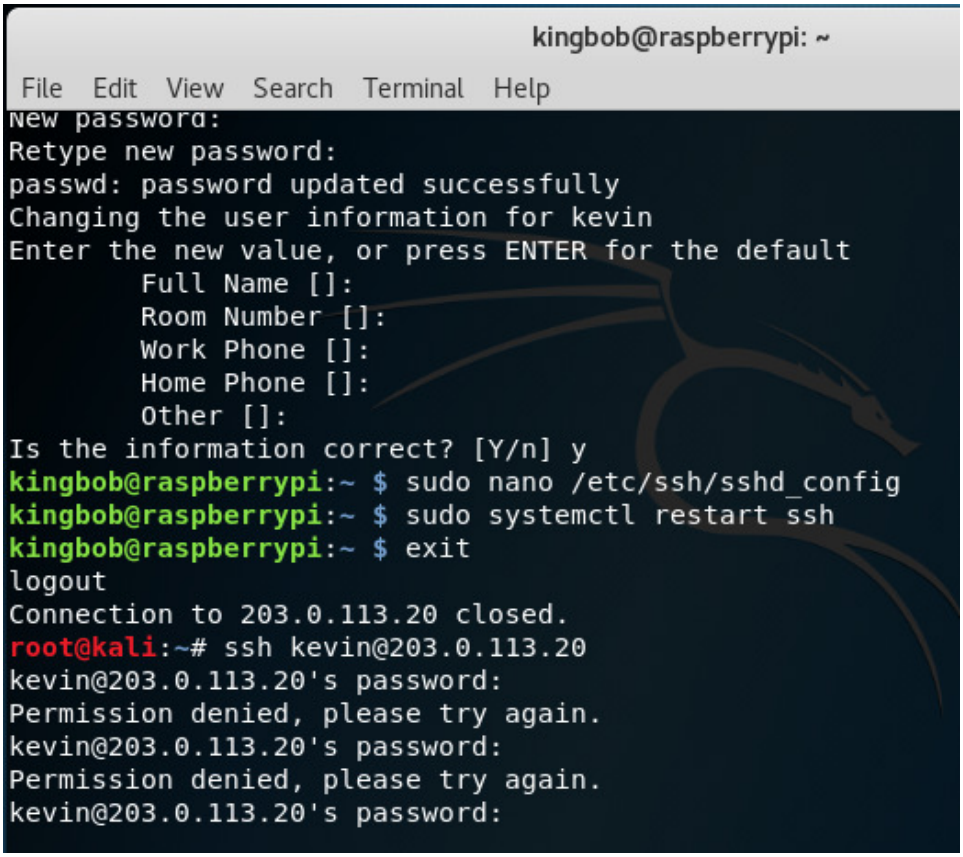
```
root@kali:~# ssh kevin@<<Pi's IP address>>
```

g. After issuing the password for the **kevin** user account the Pi reports **SSH** access is denied.

```
kevin@<<IP address of the Pi>> password:
Permission denied, please try again.
```

Press **Ctrl-C** to cancel the command

h. Take a screenshot of the terminal window at this point



i. **SSH** as the Pi user by typing:

`root@kali:~# ssh pi@<<Pi's IP address>>`

and try the password **raspberry**.

j. You should get a Permission Denied, because the user pi was not included in the **AllowUser** list in Step 3c above. Press **Ctrl-C** to cancel the command.

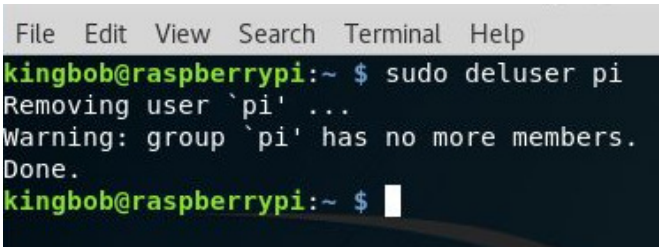
Part 2: Removing the Default Pi User Account

Step 1: Open a terminal and remove the Pi account but leave the directory.

a. To remove the pi account from the Pi, **SSH** back into the "**kingbob**" account. Then, enter the following command in the terminal.

```
kingbob@raspberrypi:~ $ sudo deluser pi
```

b. Take a screenshot of the terminal window at this point



Step 2: Require a password with the command sudo.

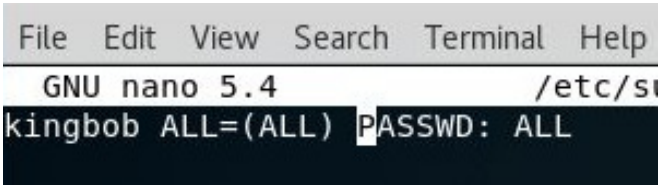
By default, the Pi OS does not require a password when placing **sudo** in front of a command to run it as a superuser . If your Pi is exposed to the Internet and somehow becomes exploited (perhaps via a webpage exploit for example), the attacker will be able to change items that require superuser rights … unless you have set sudo to require a password.

a. To force sudo to require a password, you will edit the file /**etc/sudoers.d/010_pi-nopasswd**.

```
kingbob@raspberrypi:~ $ sudo nano /etc/sudoers.d/010_pi-nopasswd
```

Replace the pi entry, **`pi ALL=(ALL) NOPASSWD: ALL`** with the username, **kingbob**, and change the option from **`NOPASSWD`** to **`PASSWD`**

```
kingbob ALL=(ALL) PASSWD: ALL
```



b. Now save the file by pressing **Ctrl-O** then enter and **Ctrl-X** to exit

c. Reboot by enter the following:

```
kingbob@raspberrypi:~ $ sudo reboot
```

d. Wait about 2 minutes and **SSH** back into **kingbob's** account. If you get the error message about no route to host, repeat the network adapter reset from Step 1b.

e. Type the following command:

```
kingbob@raspberrypi:~ $ sudo systemctl list-machines
```

You should be asked for **kingbob's** password.

f. Press **Ctrl-C** to cancel the command.

g. Take a screenshot of the terminal window at this point

Part 3: Configuring the iptables Firewall

iptables has handled the firewall configuration for Unix/Linux for a long time Here you will configure it to limit dangerous protocols and services, while allowing important connections to be available.

You will also use **nmap** (network mapper) to rapidly scan the network. **nmap** is THE network scanning tool that allows you to discover network hosts and resources, including services, ports, operating systems, and other fingerprinting information. Nmap **should not** be used to scan networks without prior permission. The act of network scanning can be considered a form of network attack.

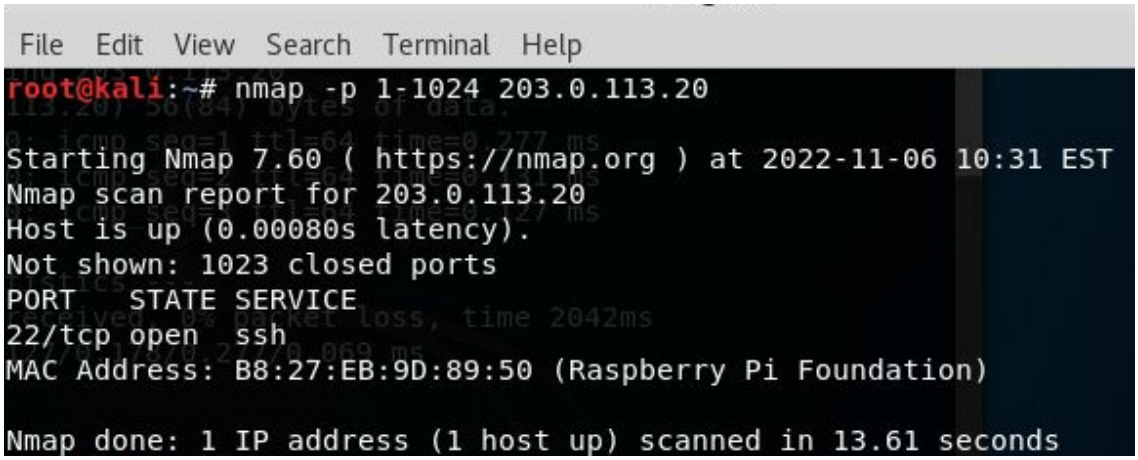
nmap will test the firewall port/service restriction and IPS capabilities of the Pi. You will run the scanning program from the Kali Linux VM and attempt to scan open ports on the Pi before and after viewing the **UFW** rules.

Step 1: Identify what network services are listening on the Pi.

a. Connect to the **Pi** over **SSH** as **kingbob**

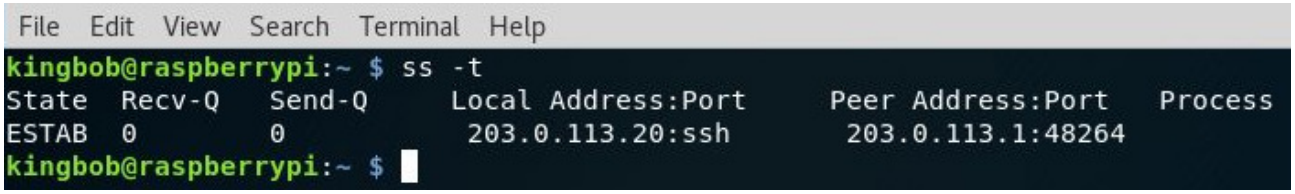
b. Open a new **Terminal** session by clicking of **File** then **Open Terminal** and perform an nmap scan targeting the TCP ports of the Pi. This scan checks all TCP ports in the range of 1-1024.

root@kali:~# nmap -p 1-1024 <<Pi's IP address>>



c. On the Pi, the currently open **TCP** sessions can be viewed by using **ss -t**:

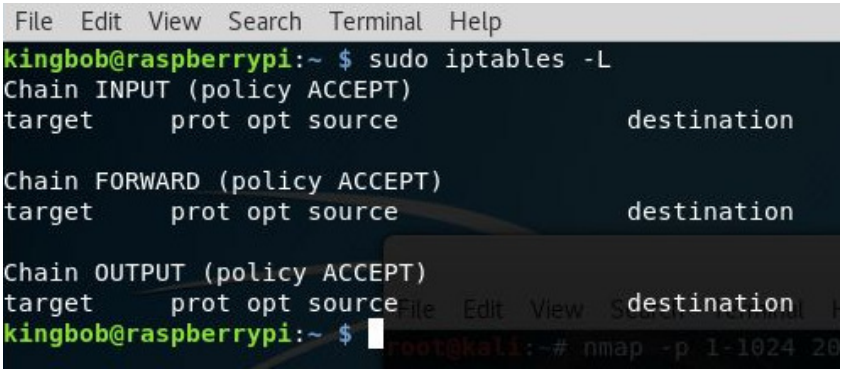
kingbob@raspberrypi:~ $ ss -t



Step 2: Check the status of iptables.

a. Type the following command to see the listings in **iptables**.

kingbob@raspberrypi:~ $ sudo iptables –L



Step 3: Configure firewall rules

At this time there are no firewall rules configured which means all traffic may pass in/out of the Pi. It is important to block all traffic, except for traffic that you want to allow.

a. If you now dropped all inbound traffic, inbound traffic not explicitly permitted will be dropped and you will lose the SSH session and be locked out of the Pi. You will first create a policy that allows **SSH** traffic into the **PI**, and then you can drop all other inbound traffic.

Type the following command to allow SSH inbound traffic:

kingbob@raspberrypi:~ $ sudo iptables –A INPUT –p tcp --dport ssh –j ACCEPT

where: **-A** indicates direction (INPUT / OUTPUT / FORWARDED)
-p indicates protocol (TCP / UDP)
-- dport indicates the TCP/UDP port or service (SSH or port 22)
-j indicates the action (ACCEPT or DROP)

b. Now you can drop all non-SSH inbound traffic by typing this command:

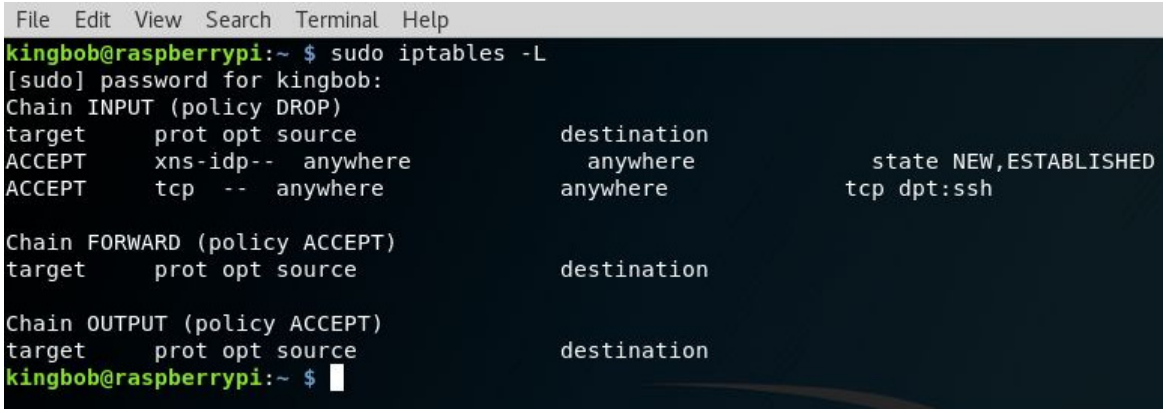
kingbob@raspberrypi:~ $ sudo iptables –P INPUT DROP

The **-P** chain target sets the policy for the built-in (non-user-defined) chain to the given target. The policy target must be either **ACCEPT** or **DROP**.

c. View the **iptables** rules by typing the following command:

kingbob@raspberrypi:~ $ **sudo iptables –L**

d. Take a screenshot of the terminal window at this point



This barely scratches the surface of how firewalls are setup and used.

Step 4: Run nmap and set scanning options.

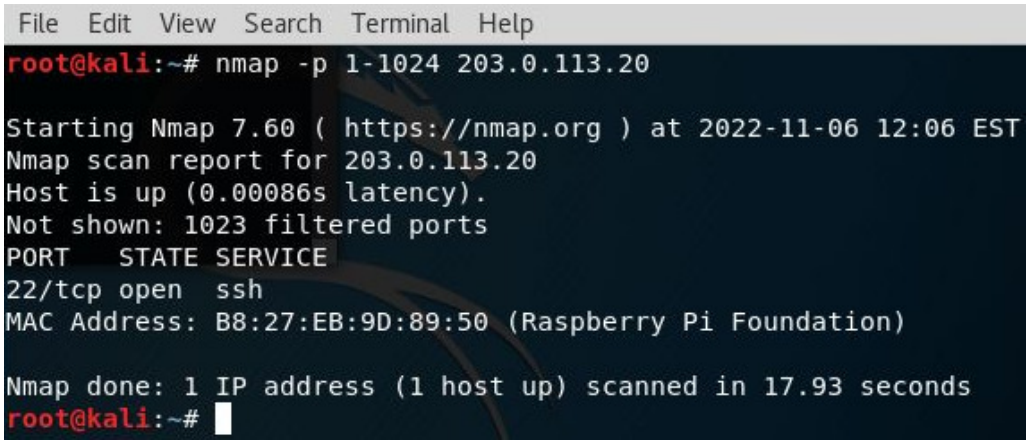
a. Go back to the Kali Linux terminal session

b. Perform an **nmap** scan targeting the TCP ports of the Pi.

`root@kali:~#` **`nmap -p 1-1024 <<IP address of PI>>`**

c. You should only see the SSH port 22 open.

d. Take a screenshot of the terminal window at this point



Submit all screenshots to the assignment page in Canvas.