Exploring Security Flaws of Unix Systems and the Basics of Hardening a Unix System

By: Mikey Joyce

Pawprint: MPJYKY

Course: UNIX Operating Systems

Topic:

In this paper my topic is all about security with Unix based systems. I chose this topic because the computing field continues to expand and has no sign of slowing down. Since everyone is using all types of computers, security will become more and more of an issue as the number of fronts an attacker can attack on keeps expanding, especially with loads of IoT devices flooding the market now. As long as the computing field is booming the security field will also boom. I wanted to use the Unix skills that I learned in this class to explore possible security risks with Unix based systems and the basics of how to harden a Linux distribution.

Research:

In this section we will be covering security risks with Unix based systems. There are four basic categories of attack that an attacker can use to try and breach a system. These categories are physical security attacks, social engineering, dumpster diving attacks, and network based attacks.

A physical attack is when the attacker has access to the physical computer. If your computer is equipped with USB ports, CD drives, etc., then chances are the device can be breached through these methods. For instance, if the computer is plugged into the ethernet an attacker can unplug the ethernet chord and plug it in his laptop to record packets on the network. The attacker can later analyze the packets to find valid login credentials. Another example of a physical attack would involve the use of a USB stick. If the attacker plugs in the USB stick into the target computer the attacker can wreak havoc on the system and obtain information such as: taking remote control over the device, access to the webcam, microphone, and keyboard, steal passwords and personal information, and much more.

The next category of attack is social engineering. Social engineering is when the attacker uses deception to manipulate individuals to obtain information that they shouldn’t have access to. Most of these attacks are designed to play on human emotions so the attacker can get what they want. For instance, say you get a voice mail that says you’re under investigation for tax fraud and you must call right away to prevent arrest. The attacker would then ensue to have you hand over personal information over the phone. This type of attack plays on the human emotion of fear. Now that companies understand the importance of customer service, attackers posing as a customer can obtain loads of information from the victim’s social media page to be able to answer basic security questions. This would make them appear as a legitimate customer if they call customer support, they can attempt to use social engineering to change the victim’s login credentials and obtain access to their account.

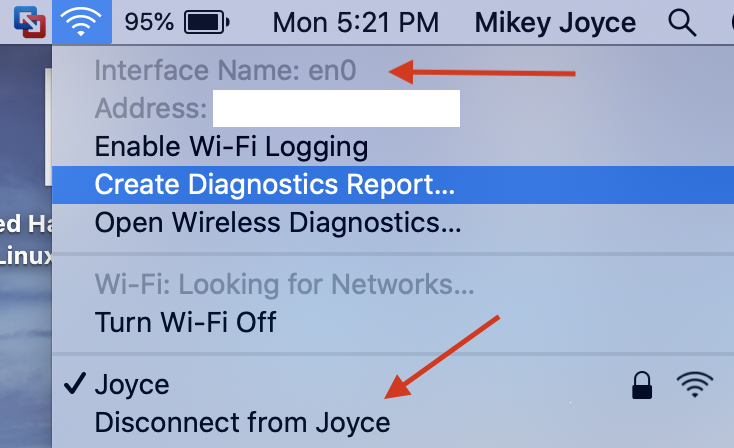
Another category of attack would be dumpster diving attacks. This is when the attacker goes through the company trash to try and obtain valuable information such as old hard drives, paper with useful information on it, and more. This type of attack is quite easy to prevent though. If the company implements policy such as shredding all paper and securely disposing of hard drives (wipe the information first), then the company won’t have to worry about this type of breach.

The fourth category of attack is network based attacks. A computer that is connected to the internet allows it to be attacked from anywhere in the world. Network based attacks are designed to compromise network security or to intercept and possibly manipulate network traffic. Three common types of network attacks are sniffing, spoofing, and Denial of Service attacks. Sniffing attacks are when the attacker gets into the network stream and is able to capture packets of data flowing between a client and server. If the data is unencrypted the attacker has the potential to be able to steal account credentials. A popular tool that allows this type of attack to happen is “tcpdump.” Another form of network based attacks are spoofing attacks. Spoofing attacks are when the attacker pretends to be a legitimate entity. For instance, a computer spoofing an IP address, Address Resolution Protocol, or Domain Name System server. If the attacker uses these types of spoofing attacks they can pose as a legitimate user and gain access to networks that are supposed to be off limits. The third network based attack is Denial of Service attacks. The Denial of Service attack is when the attacker blocks an organization or users from using its own resources such as network bandwidth or system resources (the physical computer). Typically, these attacks are carried out by the attacker flooding the network with load of junk packets or authentication requests. A Distributed Denial of Service (DDoS) attack is the same thing as the Denial of Service attack except there will be multiple attack points sending in loads of junk packets.

Now that we know lots of forms of attack that attackers may use to breach a system, we must apply the information in terms of Unix by setting up basic defense for our Linux distribution of choice. It is more important than ever to be on alert for cyber-attacks because of the expanding field of computers.

Application:

In this section I will use my knowledge of Unix operating systems to try and harden a Linux distribution to make it harder to breach. I will be demonstrating the hardening techniques specifically against network attacks. I will be using the RHEL VM that we have been using all year.

The first step I want to take will be to spoof our MAC address to hide the identity of our computer. Initially I wanted to do this on the RHEL Virtual Machine but for some reason when I was configuring my network it wouldn’t allow me to isolate the Wi-Fi on RHEL it would only allow me to share the Wi-Fi from my MacBook via an ethernet. This is alright though because I will demonstrate how to spoof the MAC address on the MacBook since the RHEL virtual machine will be sharing the same network. So, the first step will be to figure out the Wi-Fi interface name on the MacBook which will be possible by holding down the option key while you click on the Wi-Fi sign in the upper right corner of the display. The Wi-Fi interface name should be the first one down. My interface is named “en0.” You will also want to take note of the current MAC address which is located right below the interface name. Then, before clicking away you will want to disconnect from the current Wi-Fi that you are connected to.

After that we will want to open up the terminal in MacOS. The first thing we want to do in the terminal is verify the existing MAC address by running:

**$ ifconfig en0 | grep ether**

The address that is output from this command should match the address from the screenshot above (It’s the blank white space I decided to strike out the address). Next we will want to generate a random hexadecimal number that will serve as the new MAC address. After it is generated we will copy the last 3 octets (XX:XX:XX:YY:YY:YY) only copy the Y’s and then keep the X’s the same as the original MAC address:

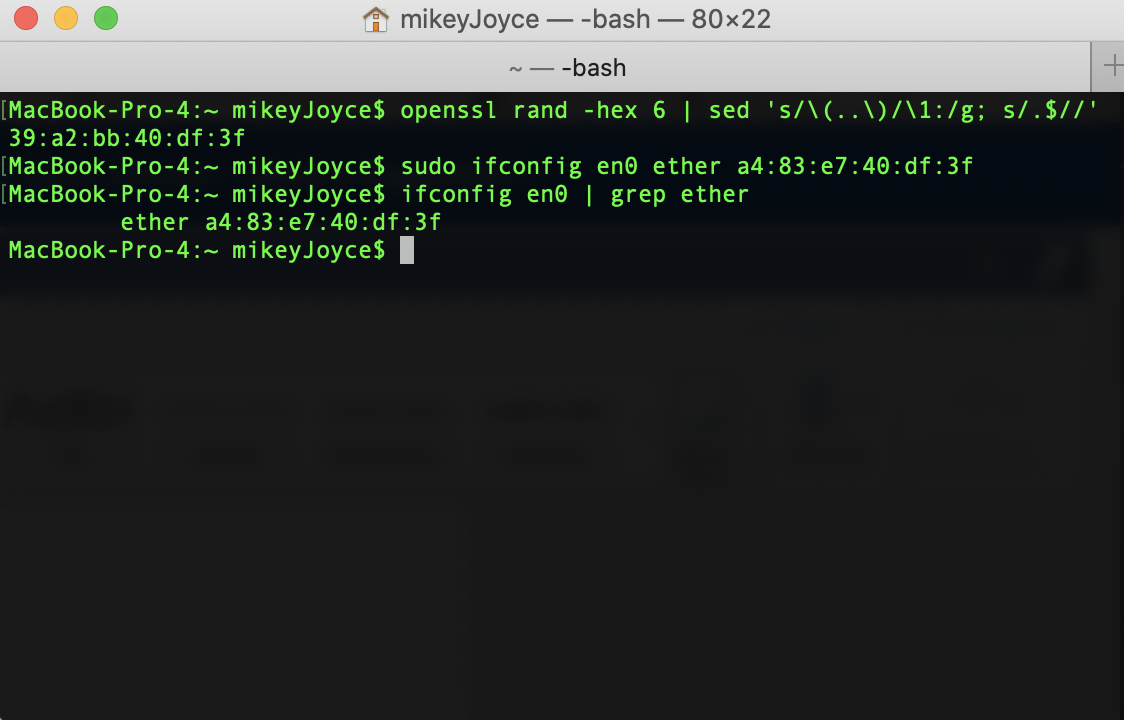
**$ openssl rand -hex 6 | sed ‘s/\(..\)/\1:/g; s/.$//’**

Then after we get the new MAC address we will run the next command to change it:

**$ sudo ifconfig en0 ether XX:XX:XX:YY:YY:YY**

You can confirm the MAC address is changed by running:

**$ ifconfig en0 | grep ether**



Now we will switch over to our RHEL Virtual Machine. We will want to look at what our computer is listening to, AKA a connection from another device to our computer designed for interaction between the two. To view the background processes in a listening state we will run:

**$ sudo netstat -ntpul**Text

Description automatically generated

You see in the right column there are some devices called “XXXX/cupsd.” Well these are printers and I have never and will never use a printer on my RHEL Virtual Machine, so this is unnecessary. To disable this run:

**$ sudo systemctl disable cups-browsed**

Next you can see another program called “XXXX/avahi-daemon.” This implements Apple’s Bounjour architecture which registers local IP addresses and static services. On my machine this is not active, but if it is active on yours you can disable it by running:

**$ sudo systemctl disable avahi-daemon**

You can analyze the rest of the listening services on your device and determine whether or not it is necessary, then you can disable the service like I have done above. Now that we have taken care of some listening service abuse that could cause a vulnerability on our machine, we will move on to defending against port abuse. Attackers may try to siphon data from our ports, so we will use a firewall to help protect our system. First we will need to install the firewall on our RHEL Virtual Machine. To do this we run these two commands:

**$ sudo dnf install firewalld**

To enable the firewall we run:

**$ sudo systemctl enable firewalld --now**

To check the status to make sure the firewall is running run:

**$ sudo firewall-cmd –state**

If it says “running” then it should be properly installed and working. Generally if we really wanted to lock up our ports and only allow the traffic we need we would just end up blocking everything and then taking an inventory and only allowing the traffic we need. This may cause some apps to react poorly and to get it right would take loads of time. Because of this I will show you the basics of how to set up our own zone with som basic firewall rules. First we will want to list which zones we already have:

**$ sudo firewall-cmd --get-zones**

This will output the already existing zones. If you want to see everything that is unblocked in a specific zone we would run:

**$ sudo firewall-cmd --zone (name of zone) --list-all**

Now we will create our own zone so that we can start our firewall customization from scratch. To create a firewall of name “test” and to activate it we will run:

**$ sudo firewall-cmd --new-zone test –permanent**

**$ sudo firewall-cmd --reload**

Once we get “success” for both then our firewall “test” has been created and is now active. Before we do anything else to customize our firewall we will allow for the SSH port to be permanently open so that we can access remotely if we wanted:

**$ sudo firewall-cmd --zone test --add-service ssh --permanent**

Next we are going to want to find the network interface that we are going to want to protect. We can do this by running and choosing the interface we want:

**$ ifconfig -a**

Text

Description automatically generated

Then after we have all of this set up, we are going to configure the interface that we are going to protect by running. I will be protecting “ens160”:

**$ sudo firewall-cmd --change-interface ens160 --zone test --permanent**

Then we will want to set the “test” zone as default:

**$ sudo firewall-cmd --set-default test**

To view the zones assigned to each interface we will run:

**$ sudo firewall-cmd --get-active-zones``**

Text

Description automatically generated

Now we have a firewall set up that blocks everything except an SSH from happening. Now if we want to customize further we should look at the list of available services that can be accessed by running:

**$ sudo firewall-cmd --get-services**

Text

Description automatically generated

This list is pretty extensive, but I will give an example of how we can allow some services to run. Since we are doing security we are only going to want to allow services that are necessary, so say we are running a web server. We will need the “httpd” service to be allowed to run. So we will install httpd and enable it with the following commands:

**$ sudo dnf install httpd**

**$ sudo systemctl start httpd**

**$ sudo systemctl status httpd**

Now we will test the webserver locally:

**$ curl --silent localhost:80 | grep title**

Text

Description automatically generated

Now we must unblock the httpd service from the firewall, so connections are allowed through. To do this we will run:

**$ sudo firewall-cmd --add-service http --permanent**

**$ sudo firewall-cmd --reload**

If you wanted to remove the http service you could do so by running:

**$ sudo firewall-cmd --remove-service http --permanent**

**$ sudo firewall-cmd --reload**

Now say you want to add and remove custom ports that you may be using. To add ports:

**$ sudo firewall-cmd --add-port (port number)/tcp --permanent**

**$ sudo firewall-cmd --reload**

And to remove ports:

**$ sudo firewall-cmd --remove-port (port number)/tcp --permanent**

**$ sudo firewall-cmd –reload**

Now that we know how to set up an extremely strict but basic firewall, we can move onto the last step that I will show for hardening our RHEL environment against network attacks. This would be to setup a VPN to defend against sniffing attacks. The VPN would allow you to hide your information from attackers or even internet service providers like AT&T from selling your information to third parties. If you’re in a country where these internet service providers censor the civilians, then it could help against that too. Some good VPNs are: NordVPN, ProtonVPN, ExpressVPN, and loads more. I would demonstrate how to work a VPN, but it does cost money so I thought I would just inform you all that it is an option.

In conclusion, we have explored some basic hardening techniques for our RHEL environment which will make it that much harder for an attacker to steal our information. This project allowed me to scratch the surface of the massive field of cyber security while also strengthening my terminal skills that I learned in this Unix Operating Systems Class. Stay safe on the internet!

Bibliography:

* <https://www.social-engineer.org/framework/general-discussion/common-attacks/customer-service/>
* <https://terranovasecurity.com/examples-of-social-engineering-attacks/>
* <https://www.vaadata.com/blog/understanding-usb-attacks%EF%BB%BF/>
* <https://cmd.inp.nsk.su/old/cmd2/manuals/unix/UNIX_Unleashed/ch12.htm>
* <https://www.ffb1.com/services-tools/security-resources/avoiding-social-engineering-attacks.html>
* <https://www.ethicalhat.com/2019/08/network-based-attacks/>
* <https://eshop.macsales.com/blog/43777-tech-101-spoofing-a-mac-address-in-macos-high-sierra/>
* <https://null-byte.wonderhowto.com/how-to/locking-down-linux-using-ubuntu-as-your-primary-os-part-2-network-attack-defense-0185709/>
* <https://firewalld.org/documentation/>