CS380 — Exercise 3

February 3, 2016

Due: Friday, February 12, 2016 before midnight (60 points)

1 Description

You should work with a partner to complete this exercise. If you do work with a partner, only one person should create the codebank project and add the other as a member with at least developer privilege.

You must create a lab report that answers all of the problems listed in this document and submit that report to `codebank.xyz` in a repository named `CS380-EX3`. You can make as many commits and push as many times as desired before the deadline.

In this exercise, you will setup a couple of virtual machines in a virtual network and perform packet sniffing while communicating over the network. You will look for some specific information passed in the packets.

2 Setting up the Virtual Machines

For this and some future exercises, we will use some premade virtual machine images provided by the SEED project. You don't have to use these images, however instructions will be given assuming they are being used.

First, download and install Virtualbox from `https://www.virtualbox.org/wiki/Downloads`. Virtualbox is *hypervisor* software that allows us to run virtual machines. Next, download the Ubuntu 12.04 VM image from the following location: `https://d9292l0q1fuir.cloudfront.net/SEED-Ubuntu-12.04.ova`.

Once you have installed Virtualbox, you can choose File → Import Appliance and choose the `ova` file. When importing, make sure you check the box labeled Reinitialize the MAC address of all network cards. After importing the machine, import a second time so that you have two copies. We will be communicating between them.

After importing both times, you should have two virtual machines listed named `SEED Ubuntu 12.04` and `SEED Ubuntu 12.04_1`.

Next, we need to set up the network our VMs will use. Go to File → Preferences → Network → NAT Networks and click the plus button on the right side to create a new NAT network. If you want, you can also click the screwdriver button to change the network settings such as changing the CIDR range, but this should not be necessary.

Next, right click on each VM and go to Settings → Network → Adapter 1 → Attached to → NAT Network and the name should automatically get populated with the network we created in the previous step. Both VMs will now exist in the same virtual network.

Finally, you can now boot up each VM by double clicking them or choosing Start on the tool bar. After booting, you should see a login screen. The username is `seed` and the password is `dees`. The root password is `seedubuntu`.

3 Packet Sniffing

3.1 Problem 1: Verifying the Network

Once both machines are booted, we should verify that the network settings are correct. Open a terminal on each machine and enter the following command:

```
$ ifconfig
```

This command will list the network interfaces on the system. You should have at least two: a loopback interface and an Ethernet interface.

Document the results of the `ifconfig` command for each VM. You can provide a screenshot or list the output. Specifically, note the interface name (something like `eth1`) and inet address.

On each machine, the inet (IPv4) address associated with the Ethernet interface should be different because they were each allocated a unique address on the virtual network. If they are the same, you need to go back and fix your network configuration. Similarly, the HWaddr (MAC address) should be different for each machine. If they are not, you should shut down the machines, go back to the network settings, and reinitialize them to new addresses.

Next, to verify we can communicate, from the first machine enter the following command:

```
$ ping -c 5 x.x.x.x
```

where `x.x.x.x` is the IP address of the second VM. If the network is configured properly, you should see 5 ICMP ping packets listed as they are sent to the second VM. The RTT is also displayed. You should receive 0% packet loss in the statistics at the end.

From the second VM, do the same thing but ping the first VM's IP address. Document the output of both `ping` commands.

3.2 Problem 2: Writing a Packet Sniffer

Instead of using Wireshark, we will look at how we might write our own packet sniffer using the pcap library. Read through the tutorial at `http://www.tcpdump.org/pcap.htm` and briefly summarize how the pcap library is used. You don't have to get too specific about the code, especially if you have not learned much about C or C++ programming yet.

On one of the VMs, download the `sniffex.c` program listed at the bottom of the tutorial. You may have to shutdown the VM and switch the network settings back to NAT temporarily so that it can access the Internet. Open a terminal and compile it using the following command:

```
$ gcc -o sniffex sniffex.c -lpcap
```

Try running the packet sniffer using:

```
$ ./sniffex ethX
```

where `ethX` is your network interface. What happens? Why does the program fail to start properly?

We can solve this by running it with administrator privileges:

```
$ sudo ./sniffex ethX
```

Test the packet sniffer program by listening to the interface you looked up in problem 1. While your packet sniffer is running on one machine, ping that machine from the second one. Document the results.

Now, modify the filter expression in your packet sniffer source code so that it only captures TCP packets. Redo the above experiment and document the results. What has changed?

3.3 Problem 3: Password Sniffing

On the machine where you are running the packet sniffer, start the telnet service using:

```
$ sudo service openbsd-inetd start
```

Telnet is a service that allows a user to login to a system remotely.

After starting the telnet service, from the other VM run the following command:

```
$ telnet x.x.x.x
```

where `x.x.x.x` is the IP address of the machine running the telnet server. After some time, you should see a login prompt. Enter the login credentials of the user on the first VM and you will be logged in.

`cd` to the user's Desktop folder and create a new text file, then use the `exit` command to logout. From the first VM, verify that the text file has been created and document all of these actions.

Now, on the machine running the telnet server start your packet sniffer. You may need to modify it to sniff more than just 10 packets. Then repeat the process above from the second VM. Pay attention to the packet contents in the packet sniffer and you should find the password of the user logging in. Document the results.

Finally, repeat the process again but use Wireshark instead of your own packet sniffer. Can you still locate the user's password? Document your results. You should go to File → Save As... and save the capture file from Wirehark as `problem3.pcap`. Submit it with your report.

After these experiments, discuss your thoughts about the security of using `telnet` as a method of remotely using a system.

3.4 Problem 4: SSH

From the machine where you were running your packet sniffer, have Wireshark listen on the network interface. From the other machine, we will now login using SSH (Secure SHell). SSH is like telent in that it allows us to remotely login to a system. However, SSH was designed to be secure and uses encryption to prevent eavesdropping.

To login, use the following command:

```
$ ssh seed@x.x.x.x
```

where `x.x.x.x` is the IP address of the other VM (where wireshark is running). You might get a warning about an ECDSA key fingerprint, simply type yes and then Enter to continue. You should then be prompted for a password and finally be logged in to the system. After logging in successfully, you can exit.

Once you have completed these steps, go back to Wireshark on the first VM and try to locate the SSH traffic. Can you find the user's password? Document the results from Wireshark. You should go to File → Save As... and save the capture file from Wirehark as `problem4.pcap`. Submit it with your report.