Identifying WLAN Attacks Using Aircrack-ng

New Mexico Supercomputing Challenge

Final Report

April 8th, 2020

Team 14

Capital High School

Team Members:

Hansel Chavez

Jonathan Garcia

Lourdes Armenta Cazares

Isel Aragon

Erika Delgadillo

Teacher: Irina Cislaru

Project Mentor: Jason Schaefer

Table of contents:

Executive Summary:

WLAN also known as Wireless Local Area Network, has become very popular starting in the early 2000s when the internet became more prevalent. It was first introduced on September 21, 1998. WLAN is sometimes confused as WIFI, when WIFI is actually an underlying technology of wireless local area networks. WIFI uses radio signals to transmit data from one device to another while WLAN is a method for two or more devices that use high-frequency radio waves and allows users to move around the coverage area, often a home or small office while maintaining a network connection.

WLAN is at the point where hackers can easily get the network's password; with that password they could easily retrieve information from these small businesses and homes with little security. The business and home, have a mindset of “we won’t get

hacked”, in reality they are at risk of being a victim of a hacker without the proper security measures.

Aircrack-ng is one of the most popular WLAN password cracking tools used in hacking. Where with just a few simple lines of codes, you will be able to get the networks password.Though not only that, the tool can also identify devices connected to the network, disconnect a device from network, testing networks vulnerability in injections, and much more. [1]

Introduction:

In the world of technology, codes are not completely sealed because there are cracks in between that are big enough to let ‘short cuts’ get through and break in the system. These ‘short cuts’ each have a different way of functioning and have different roots, in other words these ‘roots’ are network software that are used for hacking. Yet, not all softwares is used for bad (hacking). Most of these softwares are used for protecting us against these hackers. With the easy set up of creating your own WLAN, we would like to manage the security of the device, if done wrong, we are at risk of hacker attacks [2]. This vulnerability can be seen within small businesses and homes. There are many ways that hackers can attack public wifi among them man-in-the-middle, Evil twin, passive sniffing, cowpath and even Aircrack-Ng and just as Aircrack-Ng is used to retrieve said sensitive information with simple commands within

the command prompt. Aircrack-ng can also be used to undo said things. With the use of Aircrack-ng (a set of hacking tools to identify and test, the vulnerability of the network), we can detect all devices found within our network. By using this program, we can implement the commands on a BASH script, implementing those commands within our terminal-based Python program.

Methods:

Our program is to identify and stop the connection of outside devices found within homes or business WLAN using Aircrack-ng. Running the program, will record the devices found within the WLAN every 5 - 15 seconds in a .csv file. If the program detects an outside device, it will disconnect the device from the WLAN and will notify the user with details of the device. If a user identifies the device as trusted, it will be added to a list of devices found within the program.

When running the program, we can easily leave the computer running for a period of time. In that time, we connect a seperate device to the network where the program should detect the device, disconnecting it, and setting aside the information until the user stops the program. When we stop the program, the program can easily present the device(s) it detected, informing the user of the device's detail (MAC address (the device code), location last found, etc.) and setting the option of adding the device to a list of trusted devices.

Code:

We are using `os` and `csv` modules in our program. `os` helps us run commands from the command prompt within our python program; `csv` will help us read the .csv files when looping through our program.

We enforce the user to be rooted (the user name or account that by default has access to all commands and files on a Linux or other Unix-like operating system); when the user is rooted, it will go through a diagnostic before entering in the terminal prompt. It will first check if the device has installed Aircrack-ng:

```
if os.geteuid() == 0:
   # Checking if aircrack is available
    print("Checking if 'aircrack-ng' is installed...")
    find_aircrack = os.popen("dpkg -l | grep aircrack-ng").read()
    if find_aircrack == "":
        print("Unable to find the program : installing it\n")
        os.popen("apt-get -y install aircrack-ng")
    else:
        print("Aircrack-ng was found in your computer!\n")
else:
    print("You must be in root to run this script...")
```

Then, it will prompt the user for the internet connection type, in this case it is `wlp2s0` (my wifi interface), others might be something like `en1` (the ethernet interface) and the users prefered .txt file for their list of trusted devices. Now the program will be forever looping the connection until the user cancels the loop or when the program detects the device, this is where we use BASH script:

```
    print("Enter internet connection type (i, g; wlp2s0)")
    internet = input(" > ")
    print("Enter network BSSID")
    bssid = input(" > ")

    print("Starting connection ; 'airodump-ng'...")
```

```
    while True:
        usr_input = input("\nThe program make your device go offline until you exit
out the program. Will you still like to continue? [y / n] > ")

        if usr_input.lower() == "y":

            print("Disconnecting device & listing connected devices in network...")
            print("Alright, good-night!")
            os.system(f"sh grab_devices.sh {internet} {bssid}")
            break
        elif usr_input.lower() == "n":
            print("\nProceeding to exit the program...")
            exit()
        else:
            print("\nInvalid input")
```

```
INTERFACE=$1
BSSID=$2

echo "Starting interface..."
airmon-ng start ${INTERFACE}

while :
do
    date +"%T"; echo " ] Iterating..."
    timeout 60 airodump-ng --bssid ${BSSID} -w airodump_info/detections/.
${INTERFACE}mon
    tail --line 2 airodump_info/detections/.-01.csv | cut -d ',' -f 1-1 >> BSSID.txt
    sleep 3m
done
```

When the user stops or the program detects an untrusted device, the computer will retrieve the .csv file found in the folder, and will run through a loop going through each line of text in the .txt files until it detects a threat or not, informing the user.

When all that has gone through, we now see a terminal prompt with `netDet>`, where it consists of two commands `trusted` and `detected`. `trusted` will list the devices you have trusted; `detected` will print out the device's information the program has found within this forever loop

```
    while True:
        usr_input = input("netDet> ") trusted

        if usr_input == "trusted":
            f = open(file_inp, "r")
            for i in f:
                print(i)

        elif usr_input == "detection":
            print("Here are the detected untrusted device(s):")
            f = open("airodump_info/detections", "r")
            for i in f:
                print(i)

        elif usr_input == "q" or usr_input == "exit":
            os.popen(f"airmon-ng stop {internet}mon")
            print("Connected back to the internet")
            print("You might need to reboot the computer to go back to the
network\n")
            print("Have a nice day! o/")
            break

        elif usr_input == "help":
            print("""
trusted     It will list all the trusted devices from text file
            you have entered.
detection   Will compare the list you've given over 'aircrack-ng'
            and will return the results whether or not it detected
            an untrusted device. (Still in progress...)
q / exit    Exit out the program.
            """)
        else:
            print("Command undefined, enter help for commands")
```

When the user exits out the program, they are able to connect back to the internet.

Results:

We are able to identify, and stop the connection of the device when running the program. And we are able to successfully grab information from the untrusted device, but do keep in mind, we can only retrieve the device's MAC address. However, we are unable to identify devices that are connected to a VPN (sending the device to another

network), which hides the device from the list of connected networks, resulting in not finding the device's detail. Unless we crack a specific layer, but with the risk of disrupting the WLAN network.

Conclusion:

From all the software networks that could have been chosen Aircrack-ng is the best candidate for this assessment because of its capability to abbreviate the time it takes to function and give a response. But keep in mind that this will not protect you from a device connected to a VPN. On our next version of this, we will try to retrieve the device's info, whether or not it is in VPN or not. We will also try to retrieve more information from the untrusted devices if possible, such as getting the device's last logged location, the computer username, etc.

This program has opened the possibilities of implementing other hacking tools within our program, such as nmap (a network vulnerability scanning tool) and netcat (a utility used for reading from and writing to network connections), protecting our network even better.

Achievements:

We have fully learned the use of Aircrack-ng and its tools not only to attack networks, but to protect ourselves within the situation of being attacked. This has also improved our knowledge of network vulnerability, disconnecting connections within our

network and much more. Not to mention, that this has given us an opportunity to realize that our information can be exposed to anyone who is willing to hack the system and that anyone is at risk of being a victim of their information being exposed. It has also made us realize that any software with the capacity is dangerous and powerful enough to pass barriers that should not be crossed. This has given us the entrance to the many tools found in the hacking community, such as nmap and netcat. Learning them, and possibly adding them to our program for our next program.

Acknowledgment:

We would like to acknowledge Jason Scheafer from Schaefer Consulting for the confusing rollercoaster of Aircrack-ng, understanding each detail about what is displaying in our terminal. We would also like to thank Hansel's household for using their network on experimenting and to run our program.

Resources:

[1] https://www.aircrack-ng.org/doku.php

[2] Vishal Kumkar, et al. "Vulnerabilities of Wireless Security protocols (WEP and WPA2)", International Journal of Advanced Research in Computer Engineering & Technology, 2, April 2012, https://dl.irstu.com/wp-content/uploads/Download/Education/Book/Network/Network%20Security/WEP-WPA-Article/Vulnerabilities%20of%20Wireless%20Security%20protocols.pdf