Wireless Security on Public Wi-Fi Networks

Alex Andrews, Keith Cornell, Zach Linderman, Adam Meyer,

Nathan Snyder, Thomas Wilson

CIT 46000 - Wireless Security, IUPUI

Indianapolis, Indiana, USA

***Abstract* -- The report focuses on designing and creating a Raspberry Pi v2 Model B - called a Rasp-Pi-Scanner - for researching and testing vulnerabilities and weaknesses on open Wi-Fi networks at publicly-accessible establishments. This includes installing the necessary tools on the Raspberry Pi, as well as developing an "automation of wireless information gathering" script. The purpose of the script is to collect data inconspicuously from establishments like Starbucks and Panera Bread. The goal of the project is to uncover any vulnerabilities and provide recommendations for improving them. Analysis of the data from our prototype Raspberry Pi shows unsatisfactory results. Our findings show that a better implementation is possible and could potentially provide rich data for improving open Wi-Fi at local businesses and organizations.**

***Keywords* -- open Wi-Fi, wireless security, Raspberry Pi, Kali, data gathering**

I. INTRODUCTION

Open Wi-Fi networks are relied upon by millions of people every day across the country in popular establishments like Starbucks and McDonalds. They are also an important part of many other governmental and religious organizations, like churches, schools, and libraries. Security and privacy are major concerns while connected on an open Wi-Fi network. According to a CNBC cybersecurity article on public WiFi networks, "more than 60 percent of consumers think their information is safe when using public internet [and]...only half of consumers think they are responsible for securing their information" [1]. Our group wanted to know if public WiFi can be made more secure and what people can do to better protect themselves.

Specifically, it is the policy of many business establishments to restrict access to the open Wi-Fi to a limited amount of time during peak hours [2]. It was assumed that many of the users, once kicked off the public Wi-Fi service, will immediately attempt to re-establish a connection at some point during this time. A new handshake would be initiated, which presented an opportunity for an attacker to exploit. It was within this trajectory that the group tried to emulate.

Examining the traffic generated as users try reestablishing a connection provides invaluable research to developing a more secure open Wi-Fi network. And it is possible to help educate the public as well as teach good practices with public Wi-Fi.

II. BACKGROUND

The main inspiration for the Rasp-Pi-Scanner originated from a fascinating article by Scott Hogg called *Raspberry Pi as a Network Monitoring Node* [3]. The purpose of the node was for troubleshooting remote networks with the Raspberry Pi. The group took this idea and developed and molded it into a tool for performing data-gathering research and tests on open Wi-Fi networks. We coined this idea "war walking".

The group's CIT undergraduate education and lab work in wireless capture and security concepts and theories laid the foundation for this project. Specifically, the group's work with wireless scanning tools (e.g., Wireshark, Kismet, Vistumbler) and the understanding of wireless architectures aided in the troubleshooting on this project.

Wireless security plays a significant role in the transmission process of data over a network medium. It can ensure the reliability, integrity, and confidentiality that the information being sent or received has been delivered to the anticipated device without alteration. Advancements in wireless security through the years has made a drastic change in the communication methodology over the internet. Each network can be configured to its own set of rules to allow or deny specific forms of traffic, which will

correspond with how well defended a network has been constructed. Best known practices are used by most organizations, but they aren't implemented in all cases.

For ease of access, there are businesses that still use unauthenticated open Wi-Fi networks that allow users to connect to an insecure access point. The idea behind this approach is to permit users access to free internet at their establishment. Unfortunately, this is done without really considering the potential threats that could affect the end-user. Data on an open network isn't sent in a secure format, thus putting users at risk to have their information monitored, dropped, or even altered by a malicious user. For the purpose of this project, the group wanted to determine what kind of traffic could be seen on an open network by utilizing tools that will capture data packets to all devices within relative proximity of the network adapter.

III. DATA COLLECTION PROCESS

The data collection process started with building the Raspberry Pi and creating an automated script to gather the wireless packets. Details about both the Pi setup and the script are explained in detail in the next two sections.

The Rasp-Pi-Scanner was taken to different locations by each group member. The startup of the device was simple and efficient. The scanner was booted up, and the script was easily edited with the new file name. After these quick steps, the device was placed out of sight. Our setup is small enough to fit into a backpack, so it allows for easy transport into each establishment. While the member sat at the establishment for a period of 30 - 60 minutes, the scanner was collecting data from the open Wi-Fi network.

Also, the scanner remained off the network. The monitor mode allowed the device to capture the radio transmissions of the network's access point.

IV. BUILDING THE PI

The original intent of the design was to build it with two important physical characteristics in mind: self-contained and extreme portability--something that could easily fit in a backpack or another innocuous container. This portability would allow any team member to keep it on their person without attracting unwanted attention.

The 7" touchscreen had already been purchased with the Raspberry Pi 2B by one member, and the Alfa antenna had previously been acquired for the Wireless Security class. Inspired by the Hogg article, the group created a shopping list of applications that the article had recommended. Over time and as the need arose, the group installed more applications like tshark and airodump-ng. Figure 1 shows the final results of the Rasp-Pi-Scanner.

Figure 1: Raspberry Pi's final build

Figure 2 shows the Alfa antenna that was used during the project. For convenience and ease, it was taped to the back of the Rasp-Pi-Scanner during the scans.

Figure 2: Alfa network adapter

Figure 3 shows the installation of the tools on the scanner, specifically aircrack.

Figure 3: Installation of aircrack

Also, the keyboard, seen in Figure 1, is a Bluetooth-connected, compact touch-pad. The keyboard allowed any group member to easily navigate the Debian-based Raspbian OS.

As stated earlier, a variety of applications were installed. However, not all of these applications were utilized, even though they were part of the original shopping list. For instance, Nmap was not used in the execution of the final project, as shown in Figure 4.

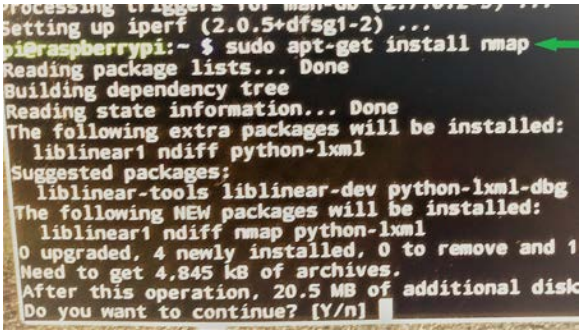

Figure 4: Installation of Nmap on the scanner

V. THE SCRIPT

The purpose of the script was to automate the capturing of data on the open Wi-Fi network. The idea was to capture data while the Rasp-Pi-Scanner was hidden in a backpack or bag. The Rasp-Pi-Scanner can draw attention because of its unusual appearance. So, this script allows for the discrete capture of data. The script can be started and then put into a bag or other discrete location.

The script consists of a few main components and was built from scratch based on what we wanted to accomplish in this project. Figure 5 shows the original script prior to any alterations.

```
#!/bin/sh

#Bash Script for automation of wireless information gathering v1.3
#IUPUI CIT 460 Semester project
#Nathan Snyder, Alex Andrews
#Assumes you have aircrack-ng suite, macchanger, and wireshark installed


echo starting wireless packet capture...
sleep 3
#change mac address to hide real mac
echo changing mac address to hide real mac
sleep 3
macchanger --mac 00:11:22:33:44:55 wlan0
sleep 3
echo hid that mac!
sleep 5
#start airmon on default wlan0, edit if different interface
airmon-ng start wlan0
#log all traffic from nearby APs -I for monitor mode -k to start capture
immediately -w for outfile
#edit after the -w for your specific capture, your name, location, and
number if more than one
sudo touch cit460_name_location_num
sudo chmod o=rw cit460_name_location_num
sudo tshark -i wlan0  -w cit460_name_location_num
```

Figure 5: Original script

This version of the script consists of hiding the device's MAC address, starting monitor mode on the main wlan interface, and saving the captured data to a file using tshark.

This script assumes that the following packages be installed prior to running it: macchanger, airmon-ng suite, and Wireshark with tshark. Sleep commands are placed between each command and comment so the user can see what is happening and all components of the script are working correctly. Each command logs its progress to the bash shell standard out.

VI. CHALLENGES

During this project, we experienced many challenges. These include the obvious technical challenges as well as some legal challenges. The scope of this project was limited due to the fact that we didn't want to make this an active attack scenario. An active attack is both legally and ethically wrong. Rather, we decided that making this a passive scanning project would better serve our purpose.

Also, passive scans have limitations, since they are not transmitting packets. The Rasp-Pi-Scanner had to wait for beacons versus actively probing to locate access points. Another limitation is that if the scanner does not wait long enough on a channel, then it might miss an access point beacon [4]. Therefore, using this passive scanning method limits what we collected.

It also limited what could be done with any data collected. For instance, the group wanted to collect information about the activities of users on an open Wi-Fi network without having any identifiable attributes. Ethically, the plan was to strip away any identifiable, sensitive, and personal information, because this doesn't pertain to the scope of the project nor is it ethical.

VII. ALTERATIONS TO SCRIPT

The specific results of the scans are discussed in detail in the following Results Section. However, the group didn't get the results they expected. After collecting only beacon frames during the first round of scans, the group made changes to the original script. Specifically, the following changes were made:

1. Removing the "macchanger" line, since no packets were sent out
2. Removing the "sleep" line

3. Using airodump-ng tool instead of tshark
4. Ensuring the interface name doesn't change when wlan0 was set to monitor mode

Unfortunately, there were no significant changes in the results of the next round of scans following these alterations. Ultimately, the group tried performing a final round of scans using a laptop and without the Rasp-Pi-Scanner.

VIII. RESULTS

The first results the group collected were disappointing. All of the packets were broadcast packets, as shown in Figure 6.

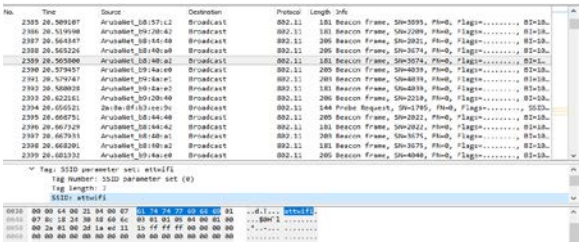

Figure 6: All broadcast frames collected

Broadcast packets do contain some important basic information pertaining to the open network, such as the SSID, channel, and associated IP addresses. Unfortunately, the large number of broadcast packets collected was not the anticipated result due to the scope of the project.

There were also a number of ARP packets that had been captured during several scans on the Rasp-Pi-Scanner. These packets are useful since they help determine IP conflicts, link bounces, system reconnects, or updating the ARP table. However, the idea behind the data collection process was to gather other forms of data than just ARP and broadcast packets. Figure 7 shows the ARP packet captures.

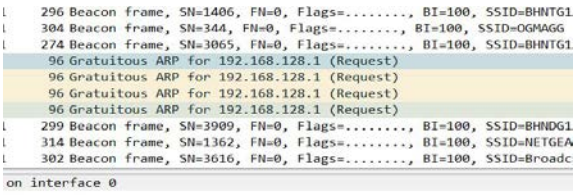

Figure 7: ARP packets

After a couple more scans, the group started making alterations to the script, as mentioned in the previous section. One of the last ideas was running some scans with a laptop instead of the Raspberry Pi. The group had an improvement in the results. To do this, Kali Linux VM was run on a laptop with the Alfa network adapter in monitor mode. The tools airodump-ng, tshark, and Wireshark were run at different times over a 60-minute period.

The group collected beacon frames, probe request frames, TCP packets, TLSv1.2 "Encrypted Handshake Message", acknowledgement (ACK) frames, and data frames. Figure 8 is an example of some of the TLS and TCP protocols.

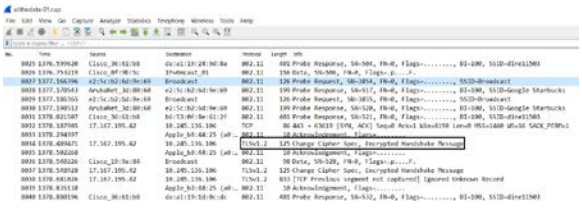

Figure 8: TLS "Encrypted Handshake Message"

Figure 9 shows an example of the 802.11 authentication packets captured.

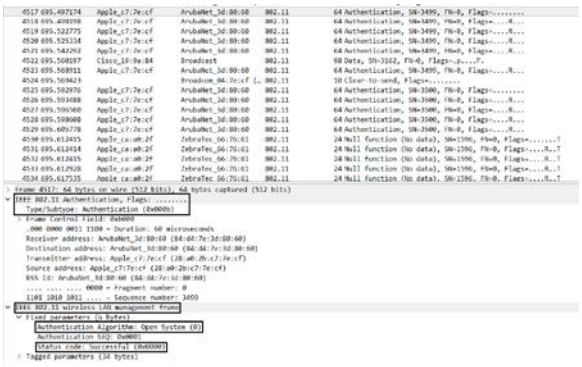

Figure 9: Authentication packets captured

Unfortunately, the TCP packets were limited in number. Further, the group wasn't able to extract any useful or meaningful information from the PCAP files. Deauthentication and association frames were also noticeably absent from these files.

Finally, we also captured a single HTTP packet. It was listed on Wireshark as a "continuation or non-HTTP traffic," which simply means that it's the continuation of content that was too large to fit into one packet [5]. The group was able to trace the IP address to a cloud-based web service. Figure 10 shows the captured packet.

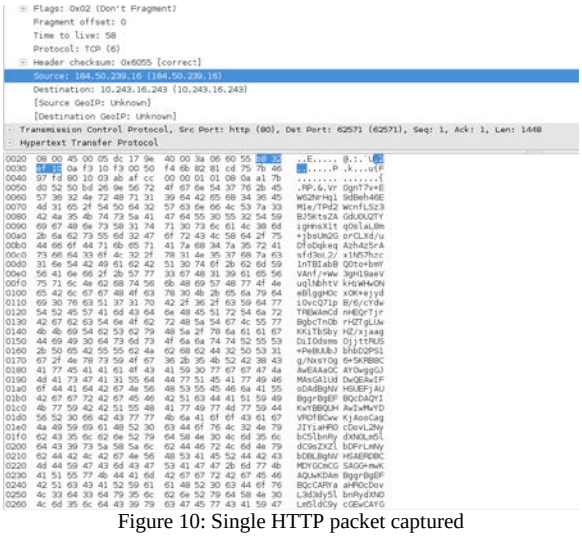

Figure 10: Single HTTP packet captured

Figure 11 shows the information found regarding this packet.

General IP Information

IP: 184.50.239.16
Decimal: 3090345744
Hostname: a184-50-239-16.deploy.static.akamaitechnologies.com
ASN: 20940
ISP: Akamai Technologies
Organization: Akamai Technologies
Services: None detected
Type: Corporate
Assignment: Static IP
Blacklist: Blacklist Check

Geolocation Information

Continent: North America
Country: United States
State/Region: Massachusetts
City: Cambridge
Latitude: 42.3626 (42° 21′ 45.36″ N)
Longitude: -71.0843 (71° 5′ 3.48″ W)
Postal Code: 02142

Figure 11: Information discovered on the packet's source

IX. CONCLUSIONS

Overall, there is a lot of room for improvement in this project. The results were not the expected aim when the project was first proposed. The group had hoped to collect meaningful data that could be statistically analyzed and presented in context to offering recommendations for improving open Wi-Fi networks.

With the current design, setup, and implementation of the Raspberry Pi and the automated script, the group only collected beacon frames. There was a significant difference when doing the same tasks on a laptop. This lead the group to consider that there was something overlooked on the initial design and setup of the Raspberry Pi itself.

One idea that was never tested was running the laptop and the Rasp-Pi-Scanner at the same location at the same time to compare the results. At this time, it is inconclusive whether this was strictly a Raspberry Pi hardware/software issue or something related to the locations of the first round of scans with the Rasp-Pi-Scanner.

There is something behind the idea that there are no failures, only learning experiences. Though the group didn't get the expected results, the project was by no means a failure. A lot was learned from this project. A major component to IT-related work is troubleshooting. The group learned how to take an idea, develop it into a working test model, and refine it. Each new alteration and test brought the group closer to a working solution. Due to time constraints, however, this solution was not attained.

X. RECOMMENDATIONS

Though the group didn't get the expected results, it was still a learning experience. The group compiled a number of recommendations for users to remain better secured while surfing on open Wi-Fi networks.

1. Ensure the connection is encrypted -- always use HTTPS protocol when browsing
2. Enable "Secure Browsing" in the security settings
3. Avoid services that are not encrypted (e.g., FTP, HTTP)
4. Avoid submitting payment and other sensitive and confidential information
5. Use a VPN (e.g., Opera VPN service)

Similar recommendations, tips, and advice are echoed on numerous online resources related to best practices on open Wi-Fi. For instance, the Department of Homeland Security gives users a non-technical list of recommendations to protecting and securing information [6]. As mentioned in the Introduction Section, the level of ignorance among users is alarming -- only half of the users think they're responsible for their own security. One of the most important things to remember is that businesses in general aren't actively protecting users' activities

on open Wi-Fi networks. It is solely up to the user to implement safeguards to protect his or her self.

XI. REFERENCES

[1] Schlesinger, J. (2016, Jun 28). *Most people unaware of the risks of using public Wi-Fi* [Online].

http://www.cnbc.com/2016/06/28/most-people-unaware-of-the-risks-of-using-public-wi-fi.html

[2] Starbucks (2017). *Starbucks Wi-Fi policy* [Online]. https://www.starbucks.com/about-us/company-information/online-policies/sdn-terms-of-use

[3] S. Hogg. (2013, Oct 30). *Raspberry Pi as a Network Monitoring Node* [Online]. http://www.networkworld.com/article/2225683/cisco-subnet/cisco-subnet-raspberry-pi-as-a-network-monitoring-node.html

[4] What are passive and active scanning? (2017). Wi-Fi Alliance [Online].

https://www.wi-fi.org/knowledge-center/faq/what-are-passive-and-active-scanning

[5] Continuation or non-HTTP traffic. (2012, Jan 12). Ask Wireshark [Online].

https://ask.wireshark.org/questions/8305/continuation-or-non-http-traffic

[6] Best Practices for Using Public Wi-Fi Tip Card [Online]. Department of Homeland Security. https://www.dhs.gov/sites/default/files/publications/Best%20Practices%20for%20Using%20Public%20WiFi.pdf