CALIFORNIA STATE UNIVERSITY SAN MARCOS

PROJECT SIGNATURE PAGE

PROJECT SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE

MASTER OF SCIENCE

IN

COMPUTER SCIENCE

PROJECT TITLE: Internet of Things: A Deeper Dive in Your Privacy and Information

AUTHOR: Manuel Maya

DATE OF SUCCESSFUL DEFENSE: May 14, 2020

THE PROJECT HAS BEEN ACCEPTED BY THE PROJECT COMMITTEE IN
PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE IN
COMPUTER SCIENCE.

Yanyan Li		5/14
PROJECT COMMITTEE CHAIR	SIGNATURE	DATE
Nahid Ebrahimi Majd		5/14/2020
PROJECT COMMITTEE MEMBER	SIGNATURE	DATE
PROJECT COMMITTEE MEMBER	SIGNATURE	DATE

Internet of Things: A Deeper Dive in Your Privacy and Information

Manuel Maya

Department of Computer Science and Information Systems

California State University San Marcos

Table of Contents

List of Abbreviations and Definitions

IoT: Internet of Things- an everyday object that has a computer embedded into it, this will allow the object to receive and send data.

ARP-Spoofing: Address Resolution Protocol spoofing- when a Hacker sends false ARP packets to the network until the malicious attacker has access to network packets from the devices connected to the router.

HTTP: Hypertext Transfer Protocol- the most basic application layer used to create websites.

MAC Address: Media Access Control Address- a unique set of numbers and letters used to identify network interface controllers associated with the device they are associated with.

IP: Internet Protocol- every device connected to the internet has one. They are used to relay information to and from the internet. Usually on your private network your router will have the IP of 192.168.1.1.

WIFI: Wireless networking technologies- WIFI can be accessed through the user's network router which will then take them to the internet.

Hacker: Malicious attacker with intent to steal or harm a user's private information stored on their electronic devices.

Raspberry Pi: Small embedded computer used in this project to receive and store network packets.

Kali Linux: Operating System ran on Raspberry Pi used to receive and store network packets. This operating system can be operated in root mode, causing very little to no restrictions for the user using this Operating System. There is also a large amount of "hacking tools" automatically installed on this Operating System.

Aircrack-ng: Networking software used to analyze, manipulate and intercept network packets from the network, this piece of software comes installed with the majority of Kali Linux systems.

WireShark: Network software used to analyze and view different types of network packets. These packets or .cap files can also show different types of network protocols as well as types of encryptions.

Scapy: A python library used for packet manipulation on networks.

Abstract

Technology is a tool used to help make societies daily lives easier. The personal computer helped pave the path for technological advancements. First came the personal computer, then smart phones came as the next major advancement and now there is the emergence of Internet of Things. Internet of Things are smaller embedded computers used to connect to a computer or smart phone. Internet of Things are used to collect, monitor, display and analyze information. Currently, of the most popular Internet of Things devices are: Amazon's Alexa Echo Dot, Nest Smart Thermometer, Google Home, Wyze Security Cameras, Smart Baby Monitors and Phillips Hue Lightbulbs. These devices all have the ability to monitor and display information through either a web portal, mobile or computer application. Internet of Things devices have been ridiculed for being in people's homes and gathering information. Although there have been security concerns about hackers breaking into these Internet of Things devices and stealing sensitive information, they are not the only security threat; big companies are also as well. Companies have been using user information for targeted ads. A large number of users have not known their information is being used for this reason. This is why it is vital for users to know what information their Internet of Things devices are capturing without their consent. This can be done by capturing network traffic via a Raspberry Pi running Kali Linux using Aircrack-ng, and then analyzing the packets using Wireshark. User information should be private and protected and no one should have their information taken from them without their consent. That is why it is also vital to have some type of monitoring device to see if a malicious attacker is trying to steal your data. Not only should this monitoring program create a database of the previous attacks that happened in the user's network, but also inform the user in real time.

Introduction

The computer has revolutionized the way society lives their daily lives. From automating tasks at work to ordering clothes and food. Internet of Things devices have been the next revolution in the technological era. Today, almost all computers and Internet of Things devices require WIFI to be used at full capacity. Without WIFI virtually all programs and features cannot be used. This is why a strong and secure connection to the internet is required. However, being connected to the internet can have its risks. When someone connects to the internet it is a two-way street. Someone can access unlimited amounts of information but can also send out information as well. That is why it is very important to have a secure connection to the internet. This way the user is aware of the information going out of their network and the information coming in. Because of this, Internet of Things devices also need to be secure. In the news, there have been countless cases of people's smart cameras, smart doorbells and smart baby monitors being hacked by malicious people. This is why security for Internet of Things devices is so vital. It is important to know what information your Internet of Things devices is being transmitted and to who it is being transmitted too.

As a result, there was an idea of a project that would capture a large amount of information being transmitted from Internet of Things devices to the network and then analyzed. The Internet of Things devices that were used were Amazon's Echo Dot, and Wyze's 1080p HD Indoor/Outdoor camera. To capture the information, a Raspberry Pi 3 running Kali Linux as the operating system. Kali Linux was installed because it is an operating system with little to zero restrictions and comes with several "Hacking Tools" installed. One of these "Tools" was Aircrack-ng a networking tool used for sniffing packets through Kali Linux's monitoring mode. Once the packets were captured, you can export them as a .cap file. A .cap file can be opened and analyzed

with Wireshark as long as you have the WIFI's password to decrypt the data captured in the .cap file. After it was analyzed in Wireshark it was exported as a .csv file where it was trained in a Python program and then generated graphs to help visualize the network traffic.

It is also important to have a program constantly monitoring and reporting your network for any malicious activity. This was done by using the Raspberry Pi 3B, Kali Linux, Scapy, a basic webpage/database, an ARP-Spoofing attack, Python 3 and saving HTTP packets. A Raspberry Pi is a single-board computer running Kali Linux, which is a Linux based operating system with several penetration and hacking tools. Scapy is a Python library used for packet manipulation, it can also be used to launch ARP-Spoofing attacks. Once the ARP-Spoofing attack is launched, it will have access to the packets of the Internet of Things device that the Attacker has targeted. Once the Attacker receives these attacks it will save them. The Monitoring Program will then monitor the network by analyzing the ARP packets being sent throughout the network. The ARP packets are then analyzed to see if the source MAC address matches the sending MAC address of the ARP packets. The Monitoring Program then raises a flag to trigger an alert to the user that there is a spoofing attack currently happening. A basic webpage was used to display the Internet of Things Name, MAC Address and the Attacker's MAC Address. It is then stored in a very simple database with previous attacks to the network. All of this was written Python and HTML.

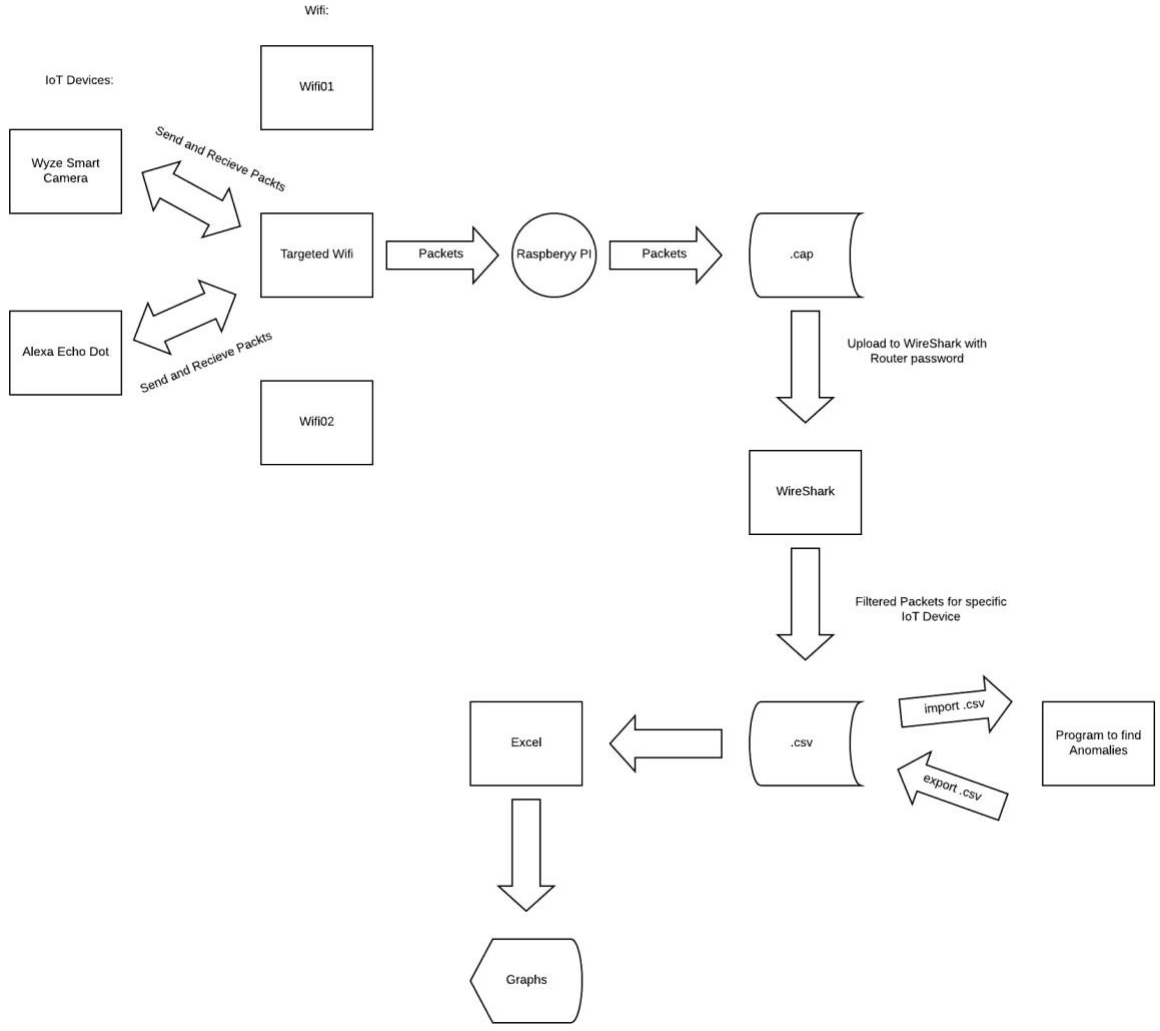
Related Work

There are several papers that emphasize Internet of Things security and the type of information that is being transmitted from the devices in your home to the devices companies' servers. The common factor in these papers are how several Internet of Things devices do not have security and how devices such as Alexa's Echo are constantly listening and sending the recordings to

Amazon's servers. Once the data recorded from the Alexa reaches Amazon's servers, they are able to use the data to create ads for the users. Amazon uses this information to generate a better user experience [5]. User data is one of the most expensive assets available in the world. Data is key for marketing for big businesses in order for them to sell more goods and products. There was also a study on how Alexa stored the user's voice and in text [3]. There are also security concerns because of how Alexa operates and stores data. One of the biggest concerns is a malicious attacker either getting close enough to use the device to get sensitive information or Alexa accidentally recording sensitive conversations and sending it to the Amazon servers [3]. While it is frightening that Alexa can send sensitive information to the Amazon servers and malicious attackers can run a man in the middle attack to obtain this information. It is quite difficult to decrypt the data being sent from Alexa. Unlike Amazon's highly secure Internet of Things encryption, the majority of smart security cameras and baby monitors have been proven to be very unsecure. The website "https://www.insecam.org/en/" is a website that live streams smart security cameras with either no password and or no encryption. The website administrator does have an ethical background and removes smart cameras being streamed from private homes and only live streams cameras in public locations (bars, streets, beaches etc.). However, if there are websites like this, there is bound to be more websites that do not have this type of ethics and will live stream people's private homes. There have been multiple cases of malicious people hacking into baby monitors. They can see what the baby is doing and even hack into the speakers and send audio through the monitor [3]. The lack of security in both smart security cameras and baby monitors are the root cause of malicious attacks, due to companies cutting cost and not adding more security and encryption [1]. Regardless, companies have a greater fault in user's data being easily stolen. If the company designed their products to have encryption, required a

secure password and had some type of two-factor authentication right out of the box, malicious attackers would have a lower probability of being able to steal sensitive user data.

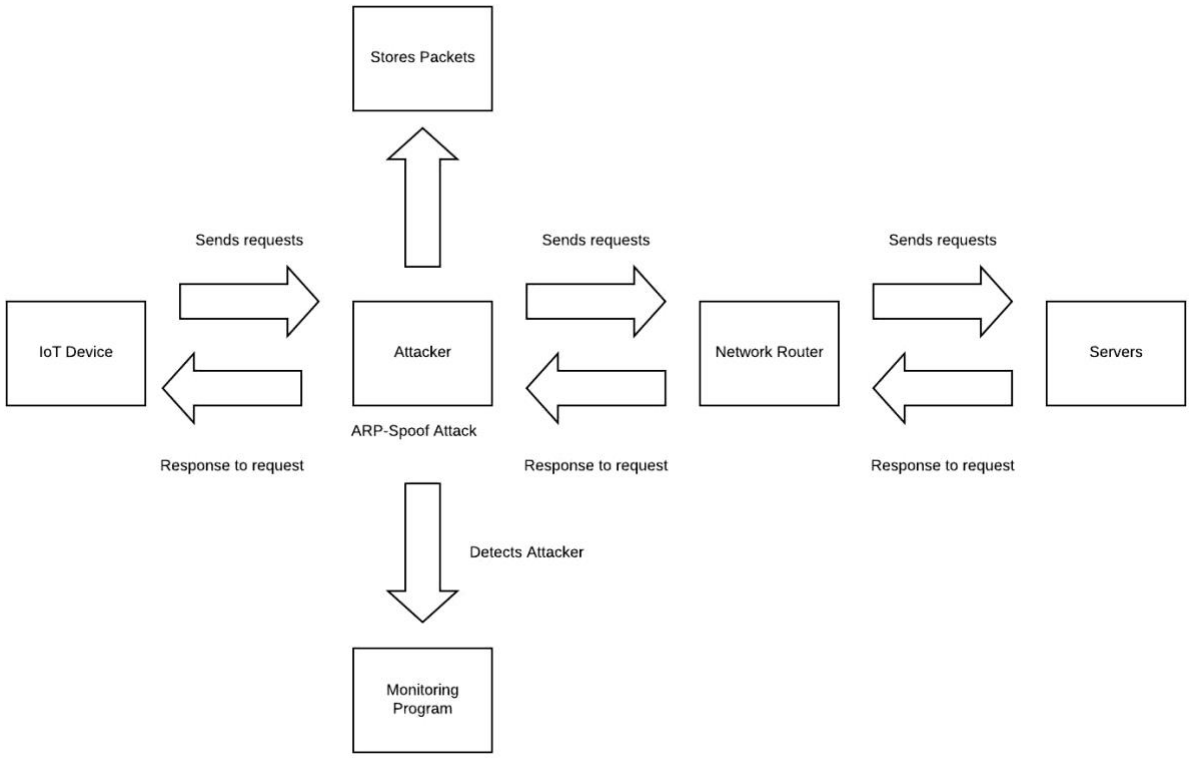
Design



Design 1

In Design 1, the design for the first part of the project is analyzing the network packets that are being sent by Amazon's Alexa Echo Dot and Wyze Smart HD Camera. The first step is having the Internet of Things device connect to the WIFI. The Raspberry Pi will then run the Aircrack-ng software to start pulling the network traffic from the WIFI router. The software then converts

the packets into a huge .cap file that can be opened in WireShark. The .cap file is encrypted because of the WIFI router. The only way to decrypt the .cap file is if the user has the router's password. WireShark then shows the packet information of the Internet of Things devices connected to the network. The user can then filter out the other network packets and only focus on the Internet of Things device that they choose to analyze. Wireshark can export the filtered packets as a .csv file. The .csv file is then imported into a program to train the data. The program then creates a new .csv file. This new .csv file is imported into Excel to create new graphs that will visualize the network packet spikes in the network.



Design 2

In Design 2, the design for the second part of the project is illustrated with a flow chart. The overall flow of the Figure 2 is a network attack called an ARP-Spoof. This attack makes the Internet of Things device believe the Attacker is the Network Router. A more detailed

description of Figure 2 is the design beginning with the Internet of Things device connecting to the Attacker. The Attacker is using an ARP-Spoof attack which mimics the IP Address of the Network Router in order to trick other Internet of Things devices into sending the Attacker their network packets. The Attacker then saves the network packets into another file and also sends the network packets to the Network Router where the packets will request the information the user wants and will get a response from the Servers. The Network Router will then relay that message to the Attacker where it will send the data back to the Internet of Things device. Because the Internet of Things devices are receiving the responses, they will not be aware that there is a malicious Attacker stealing their network packets. That is where the Monitoring Program comes in to save the day. The Monitoring Program scans the network for suspicious activity. It will grab all the ARP packets that are being transmitted in the network. The program will then analyze the MAC address of the ARP packets. The program can determine if the IP address is being spoofed if the source MAC address is different from the sender MAC address. Once the program determines that the MAC address is being spoofed, it will notify the user by updating the website. The website will display the Internet of Things Name, MAC address and Attacker's Mac Address. It will also update the database with the same information and the time and date of when the attack occurred.

Implementation

Raspberry Pi, Aircrack-ng were used to implement the packet sniffing and WireShark was used to analyze the packets going through the network. The Raspberry Pi was used because it operates with a restrictive-free operating system (Kali Linux) that allowed monitoring mode (or promiscuous mode) to be activated. Because Kali Linux has basically zero restrictions it was

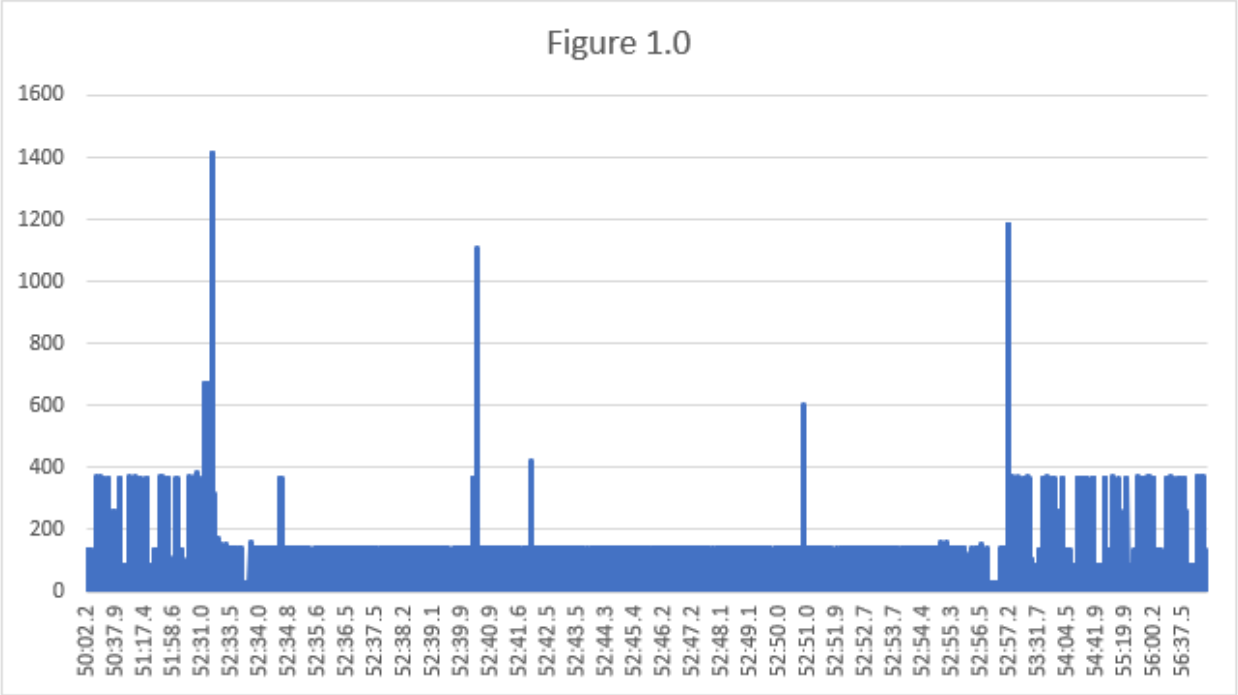
very simple to install WireShark and Aircrack-ng. Aircrack-ng is used by inputting the targeted MAC Address, Router Name and Channel Number. Without connecting to the network, Aircrack-ng will capture every packet being sent to and received to the targeted Router. Once the network packets were captured, they were saved as a .cap file. The .cap file was encrypted because of the targeted Router. To decrypt the .cap file, the file would be uploaded to WireShark then the Router password must be entered in order to decrypt the packets. Because Aircrack-ng captures everything on the network, it was important to filter the packets from the MAC address to the targeted Internet of Things device being monitored. After the packets were filtered, the results could be exported as a .csv file. This .csv file was then imported into Excel to create graphs to help visualize the packets being shared. The .csv file was also imported to a program used to train the data and find the anomalies of the captured packets. Once if finished running, it would generate another .csv file, which would once again be opened in Excel to generate another graph for visualization. By using the graphs, it made analyzing the packets in WireShark significantly easier.

The first steps for the Monitoring Program were setting up the Raspberry Pi. At first, Raspbian was installed on the Raspberry Pi as its operating system. However, after a few failed attempts to monitor the network. There was a solution that was posted in a Raspberry Pi forum. Installing Kali Linux, an operating system with no permission restrictions and several hacker tools already installed on the system. This made it significantly easier to install more hacker tools or run networking programs. After the environment was set up, there was a quick learning curve on how to utilize several of these tools. One of the best tools to use was Scapy. Scapy is a network packet manipulation software that can also be used to run ARP-spoofing attacks. This network attack sends spoofed ARP packets to trick the targeted Internet of Things device and/or the

network host. This allows the Raspberry Pi to become a man-in-the-middle network attack which allows the Raspberry Pi to intercept all the packets from the targeted Internet of Things device. At first, the intent of the project was to have an external Raspberry Pi, or another device run the ARP spoof attack. However, due to hardware issues and not having the extra Raspberry Pi. The attack and analysis came from the same Raspberry Pi device. The Internet of Things devices that we used; a Vizio Smart TV and a Roku. Scapy was used to collect the packets, which were saved in an external file to be viewed later. While all of this is happening, the ARP-spoofing was still running. Another program was written to monitor the network for suspicious activity which would then raise a flag and alert the user.

Results

The *filteredPacketsWithGraph.csv* file has the network activity of the Wyze smart cameras on the network. When the smart camera's companion app tried to view the feed of the cameras it would spike up (Figure 1.0). The program used to train the data to show which packets were higher than usual and created another csv file *out.csv* that displayed the abnormally large number of packets being sent to the company's companion app.



As displayed in Figure 1.0, the graph shows the packets being transmitted and received on the network. These packets are captured via a Man in The Middle Attack from a Raspberry Pi. This then gets analyzed using Wireshark, after the data is cleaned. It is exported into another CSV file where another graph is created to visualize the network traffic (Figure 1.1). Figure 1.0 shows how the raw network data is being transmitted for 6 minutes. 366 and 138 are the most common amount of network packages being shared through the network during this time. The graph's spikes at time 52:32.2, 52:40:4, 52:50:0 and 52:57:2. The network spikes are caused by the companion app for the smart cameras requesting a live stream straight to the mobile device. Once the app was closed, the network traffic returned back to normal.

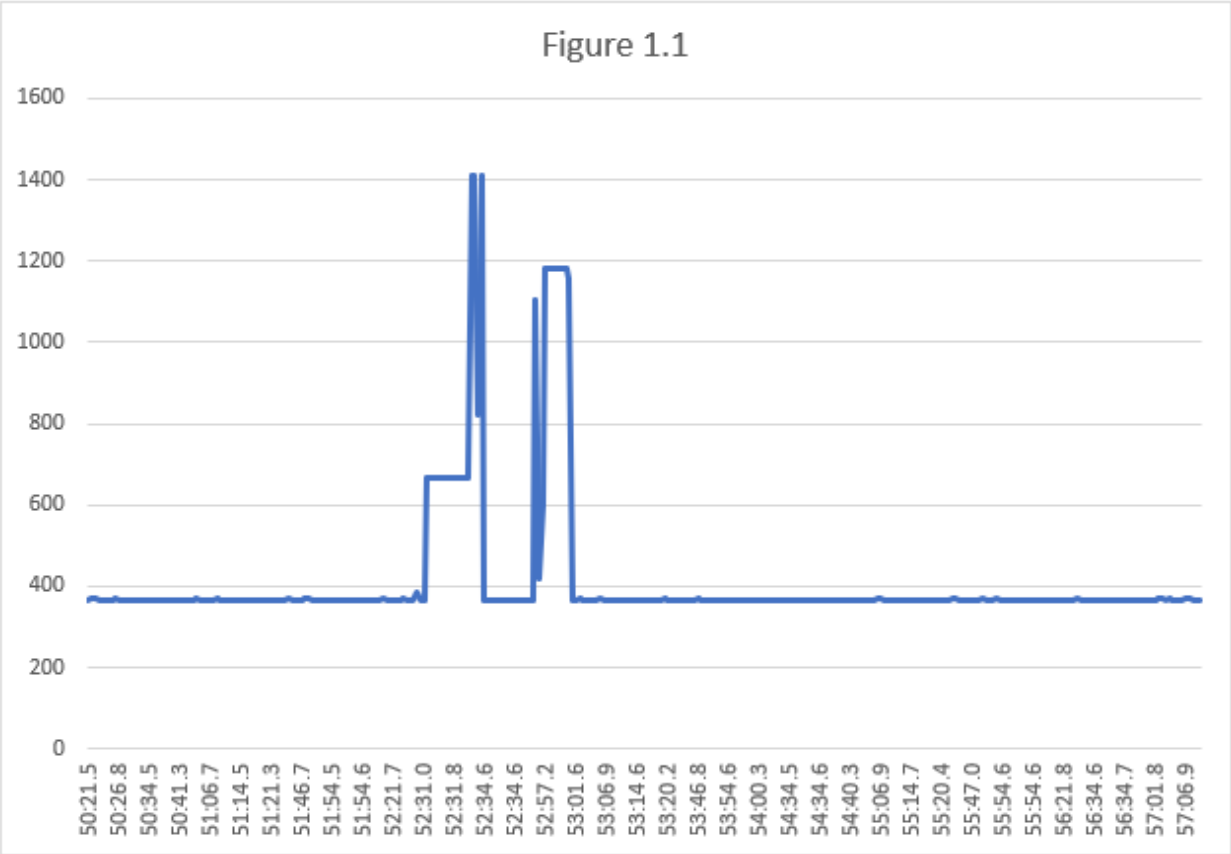
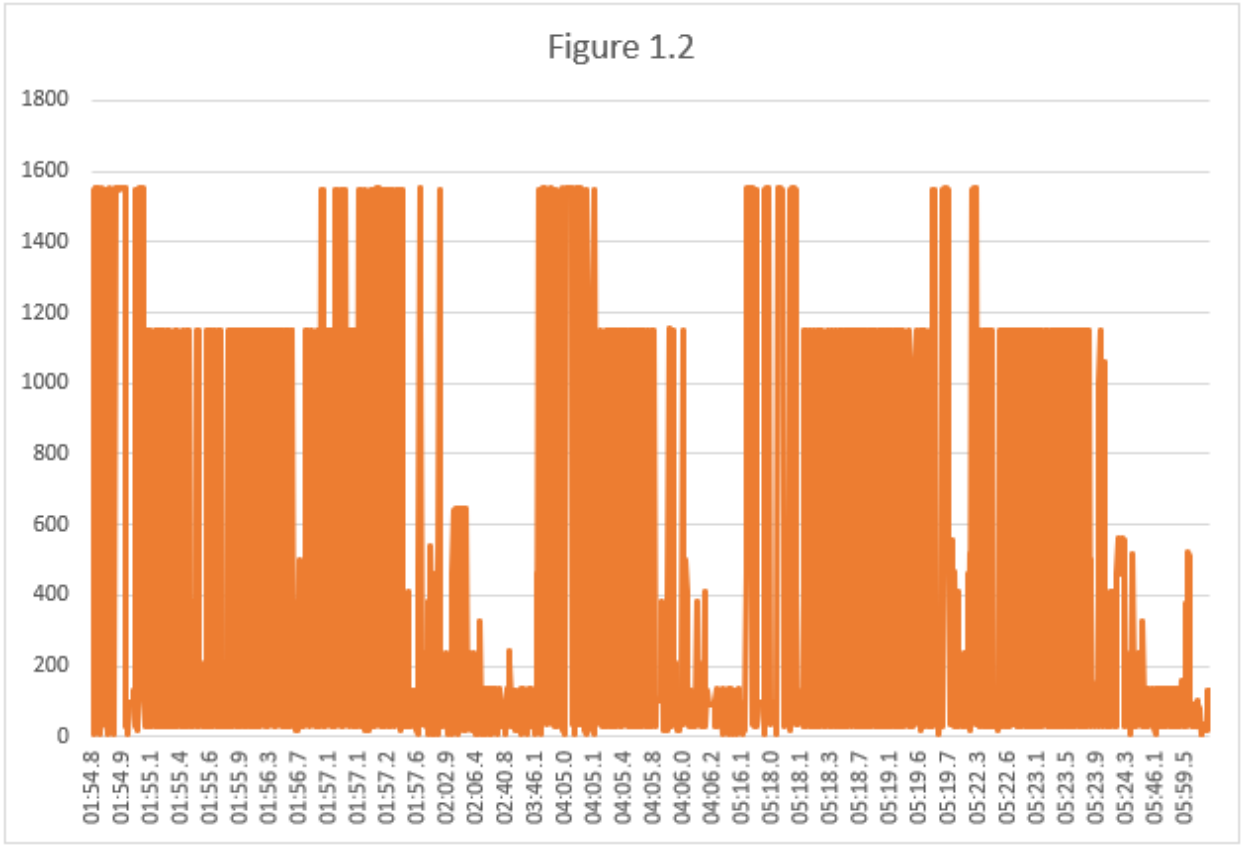


Figure 1.1 is an analysis of the network after it goes through the program that trains the data to find anomalies in the network packets. This helps visualize the anomalies by seeing the exact time it occurs so the user can go back into Wireshark to further analyze the packets.

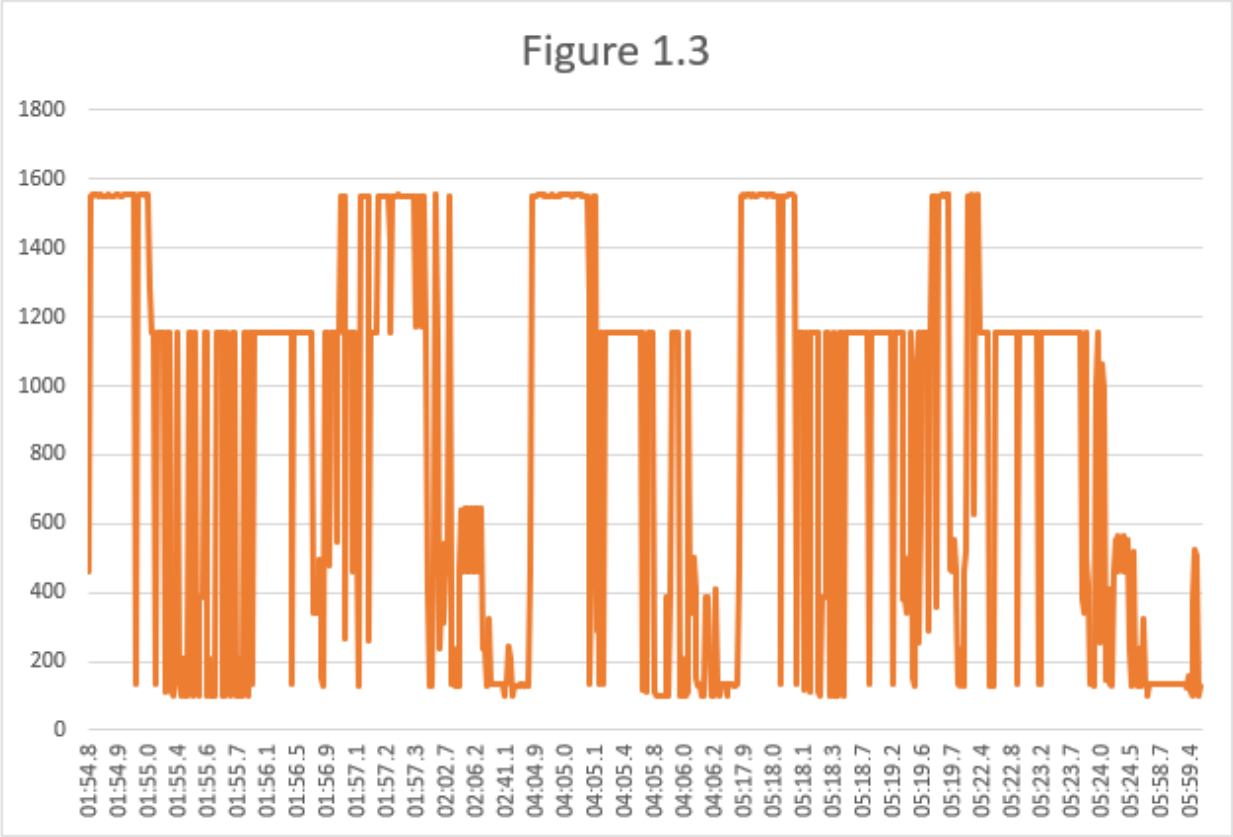
Right when the companion app requests a live stream from one of the smart cameras, there was a packet that sent a request to the smart camera named “Clear-to-Send”. When setting up the Wyze companion app, it requires you to choose a very secure password and to authenticate the Smart Cameras with the app via a security code. Once this code is paired with the app, only users with the login credentials are allowed to view the live feed. “Clear-to-Send” is the request giving permission to the companion app to view the live feed. The following packet is the Smart Camera sending a large packet (live feed) to the router to then be sent to the companion app. Not only are the packets only shared to the account linked to the Smart Cameras, but the packets are

also encrypted with CCMP. CCMP stands for Counter CBC-MAC Protocol which is an Advanced Encryption Standard. CCMP uses CCM algorithm to combine Counter Mode for confidentiality and Cipher Block Chaining Message Authentication Code for authentication [6]. Because of this, it doubles the security and makes it less likely to be hacked by malicious attackers.



The alexaOut.csv does the same but it is looking at the packets being sent when Alexa is being used. In Figure 1.2, the first 3 seconds was a command given to Alexa to play a song, then Alexa was told to stop listening, however you can see later on the packets spiked up again even though Alexa was not given a wake command to listen or play anything. To take a further look into the packets, the data was also trained to focus more on the higher spiked packets as shown in Figure

1.3. Anything after the 02:02.7 timeline was not a given a wakeup call to Alexa, which is why these packets were compared to the ones taken during the 01:54.8-01:57.3 timeframes. During non-wake but still listening timeframe, Alexa was sending Priority: Best Effort (Best Effort) (0), which means the packets will be sent but not in the highest priority.



Research began on how to better analyze and decrypt Alexa's packets being sent to the Amazon servers. WireShark was once again used to take a deeper look into the network traffic. Alexa's packets sent to the Amazon servers were sent as QoS (Quality of Services) packets. Within these packets there was a common variable of "Priority: Network Control (Voice) (7)". This means that the network prioritizes the voice packets that are being sent to Amazon's servers to a 7 (highest priority level) in order to avoid latency and packet loss. Another valuable piece of

information pulled from the packet was the "CCMP Ext. Initialization Vector: 0x000000000230". Which is also the same type of encryption that the Wyze cameras implement.

```
root@kali:~/scripts# python3 arp_spoof.py
Enabling IP Routing..
IP Routing enabled.
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
Sent to 192.168.    : 192.168     is-at b8:
Sent to 192.168.    192.168.0     is-at b8:
```

Figure 1.4

Figure 1.4 shows the ARP-spoofing attack running in a forever loop until it is told to stop. The device IPs and MAC addresses have been censored for privacy purposes. After running the ARP-spoofing attack, it was very easy to see the Internet of Things device's traffic and have it connected to the Attacker's device.

```
IP: 192.168 Requested 23.246.14.141/?o=AQP1BEL1iM-o7EayKXdAeEkZBKyBUHgNZBbIYKk
WDpFWRJ8_2kh2LYq-IlseQ2wabgJtXzg0Ss6IK1kogyEa6EigtLchvm6ML1ect_6Wbe2-Hzpt6-5Wi
M251FLiQ2a74Aux6WnbHPWLnetNtOk61jHI&v=5&e=1589277347&t=NDdJ-6oBt_eGauHzWqQ1e5o
CZW8 with GET<0x1b>[39m

IP: 192.168 Requested 23.246.14.141/?o=AQP1BEL1iM-o7EayKXdAeEkZBKyBUHgNZBbIYKk
WDpFWRJ8_2kh2LYq-IlseQ2wabgJtXzg0Ss6IK1kogyEa6EigtLchvm6ML1ect_6Wbe2-Hzpt6-5Wi
M251FLiQ2a74Aux6WnbHPWLnetNtOk61jHI&v=5&e=1589277347&t=NDdJ-6oBt_eGauHzWqQ1e5o
CZW8 with GET<0x1b>[39m

IP: 192.168 Requested 23.246.14.141/?o=AQP1BEL1iM-o7EayKXdAeEkZBKyBUHgNZBbIYKk
WDpFWRJ8_2kh2LYq-IlseQ2wabgJtXzg0Ss6IK1kogyEa6EigtLchvm6ML1ect_6Wbe2-Hzpt6-5Wi
M251FLiQ2a74Aux6WnbHPWLnetNtOk61jHI&v=5&e=1589277347&t=NDdJ-6oBt_eGauHzWqQ1e5o
CZW8 with GET<0x1b>[39m

IP: 192.168 Requested 23.246.14.141/?o=AQP1BEL1iM-o7EayKXdAeEkZBKyBUHgNZBbIYKk
WDpFWRJ8_2kh2LYq-IlseQ2wabgJtXzg0Ss6IK1kogyEa6EigtLchvm6ML1ect_6Wbe2-Hzpt6-5Wi
M251FLiQ2a74Aux6WnbHPWLnetNtOk61jHI&v=5&e=1589277347&t=NDdJ-6oBt_eGauHzWqQ1e5o
CZW8 with GET<0x1b>[39m

IP: 192.168 Requested 23.246.14.141/?o=AQP1BEL1iM-o7EayKXdAeEkZBKyBUHgNZBbIYKk
WDpFWRJ8_2kh2LYq-IlseQ2wabgJtXzg0Ss6IK1kogyEa6EigtLchvm6ML1ect_6Wbe2-Hzpt6-5Wi
M251FLiQ2a74Aux6WnbHPWLnetNtOk61jHI&v=5&e=1589277347&t=NDdJ-6oBt_eGauHzWqQ1e5o
CZW8 with GET<0x1b>[39m

IP: 192.168 Requested 23.246.14.141/?o=AQP1BEL1iM-o7EayKXdAeEkZBKyBUHgNZBbIYKk
WDpFWRJ8_2kh2LYq-IlseQ2wabgJtXzg0Ss6IK1kogyEa6EigtLchvm6ML1ect_6Wbe2-Hzpt6-5Wi
M251FLiQ2a74Aux6WnbHPWLnetNtOk61jHI&v=5&e=1589277347&t=NDdJ-6oBt_eGauHzWqQ1e5o
CZW8 with GET<0x1b>[39m

IP: 192.168 Requested 23.246.14.141/?o=AQP1BEL1iM-o7EayKXdAeEkZBKyBUHgNZBbIYKk
WDpFWRJ8_2kh2LYq-IlseQ2wabgJtXzg0Ss6IK1kogyEa6EigtLchvm6ML1ect_6Wbe2-Hzpt6-5Wi
M251FLiQ2a74Aux6WnbHPWLnetNtOk61jHI&v=5&e=1589277347&t=NDdJ-6oBt_eGauHzWqQ1e5o
CZW8 with GET<0x1b>[39m

IP: 192.168 Requested 23.246.14.141/?o=AQP1BEL1iM-o7EayKXdAeEkZBKyBUHgNZBbIYKk
WDpFWRJ8_2kh2LYq-IlseQ2wabgJtXzg0Ss6IK1kogyEa6EigtLchvm6ML1ect_6Wbe2-Hzpt6-5Wi
M251FLiQ2a74Aux6WnbHPWLnetNtOk61jHI&v=5&e=1589277347&t=NDdJ-6oBt_eGauHzWqQ1e5o
CZW8 with GET<0x1b>[39m

IP: 192.168 Requested 23.246.14.141/?o=AQP1BEL1iM-o7EayKXdAeEkZBKyBUHgNZBbIYKk
WDpFWRJ8_2kh2LYq-IlseQ2wabgJtXzg0Ss6IK1kogyEa6EigtLchvm6ML1ect_6Wbe2-Hzpt6-5Wi
M251FLiQ2a74Aux6WnbHPWLnetNtOk61jHI&v=5&e=1589277347&t=NDdJ-6oBt_eGauHzWqQ1e5o
CZW8 with GET<0x1b>[39m

IP: 192.168 Requested 23.246.14.141/?o=AQP1BEL1iM-o7EayKXdAeEkZBKyBUHgNZBbIYKk
WDpFWRJ8_2kh2LYq-IlseQ2wabgJtXzg0Ss6IK1kogyEa6EigtLchvm6ML1ect_6Wbe2-Hzpt6-5Wi
M251FLiQ2a74Aux6WnbHPWLnetNtOk61jHI&v=5&e=1589277347&t=NDdJ-6oBt_eGauHzWqQ1e5o
CZW8 with GET<0x1b>[39m
```

Figure 1.5

Because of this, the traffic from the Internet of Things devices was easily pulled and analyzed.

As you can see in Figure 1.5, the Attacker was able to pull the data from the Smart TV and find

out that the packets were going to server 23.246.14.141 (Netflix Servers). You could see that the user was sending GET requests from Netflix. The Monitoring Program would run to see if any malicious activity was in the network. The monitoring software would analyze the network traffic and look to see if there were replicas of the home network's routers IP address.

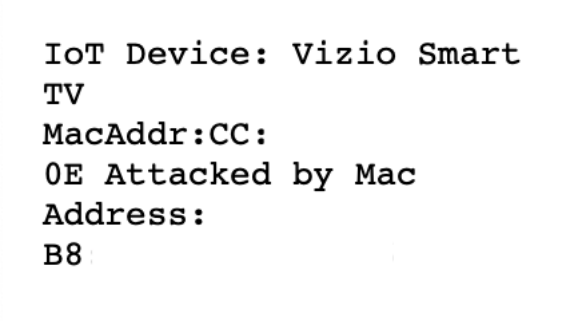


Figure 1.6

If there were any type of replicated IP addresses, it would raise an alert that the Internet of Things device's Name and MAC address are under attack by the attackers MAC address. Figure 1.6 shows how the data is displayed in the web page. The MAC addresses are censored for privacy purposes. Afterwards, the Monitoring Program would then grab the date and time of the attack and save the information into a database.

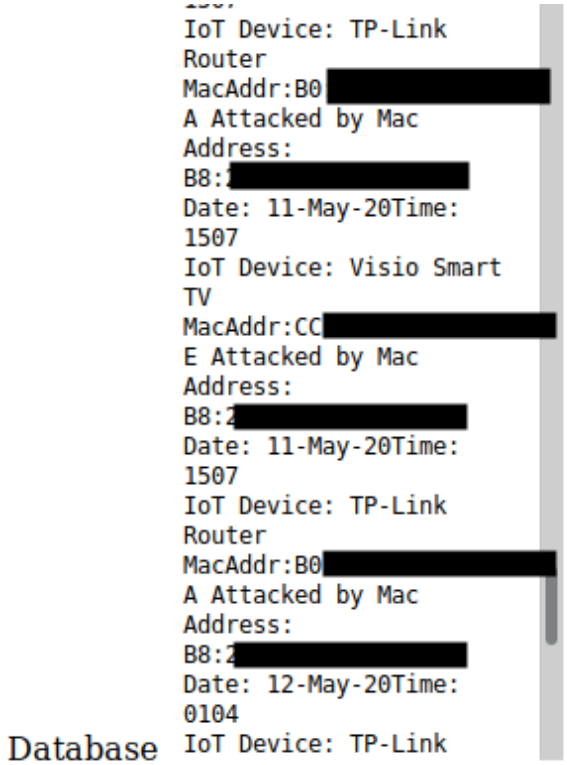


Figure 1.7

Figure 1.7 shows how the database will be displayed in the web page. The web page will first display the Internet of Things device's name that is currently being attacked. It will also display the Internet of Things device's MAC address, then it will show the Attacker's real MAC address instead of the spoofing one it was using. In the end, the database will display the date and time to see when the attacks occurred.

Conclusion

Based off the information pulled through the Man in the Middle attacks, it is possible to see when and where the packets are being sent too. However, it would be very difficult for a malicious attacker to hack into a user's WIFI (assuming they have a secure password) and being able to crack the encryption on CCMP. After consulting with Cloud, Security and Network

engineers, the main conclusion was breaking CCMP and Amazon's encryption would be nearly impossible. There are multiple teams dedicated towards keeping Amazon's security top of the notch and as secure as possible, especially with all of the sensitive data being sent from Amazon's Echo. Based off the network packets and flow, Alexa is constantly listening to user's conversations. If this information was hijacked by a malicious attacker, it would be a huge lawsuit waiting to happen to Amazon. Amazon is already being scrutinized for storing user's data without their knowledge. Amazon collects this data in order to "improve user experience" and for targeted advertisements for the users. It is important to scan your network or hide it so malicious attackers cannot steal your data. It is also important to have secure passwords and Two-Factor-Authentication. The ARP-spoofing attack was extremely simple to execute. The only thing required to execute the attack was the victim's IP address (Internet of Things device IP) and the victim's gateway IP address. Scapy is then able to run the ARP attack and start recording the victim's network packets. This is why the network's router should have a built-in monitoring system to monitor if any other device is trying to steal the gateway's IP.

Future Work

Machine Learning can be used to create nodes that search for the anomalies in the network. This can be done by analyzing the massive packet increase in the network to an Internet of Things device when it should not be sending or receiving that much information. The most difficult part of this would be how Alexa is constantly listening and sending user information to the Amazon servers. There would have to be some type of flag to trigger the nodes to let the program know that Alexa is being triggered by the user. The flag would raise an alarm if Alexa begins listening without being triggered by the user. It would also be beneficial for the program to autogenerate a

daily dashboard/report on the network traffic. Not only will this be able to help users decide on what Internet of Things devices they want in their households. But it will also give them a peace of mind for their privacy in their own home. Add more types of network attacks to this program to be able to analyze and detect them when they occur so the user can take action. It would also be beneficial to purchase a secure server to show the user's network's sensitive security information or several more Raspberry Pi devices in order to run more network attacks and monitor the data.

References

1. Mark Stanislav and Tod Beardsley “HACKING IoT: A Case Study on Baby Monitor Exposures and Vulnerabilities” *Rapid7*, 2015
2. Marcia Ford and William Palmer “Alexa, are you listening to me? An analysis of Alexa voice service network traffic” *Personal and Ubiquitous Computing*, CrossMark, January 2018
3. Hyunji Chung, Michaela Lorga and Jeffrey Voas “Alexa, Can I Trust You?” *Korea University,* National Institute of Standards and Technology, September 2017
4. Jason Beahm and Cameron Bowman. “Alexa, Are You a Snitch*?” Offices of Lawyers*, Beahm Law, May 2019
5. Anne Logsdon Smith. “Alexa, Who Owns My Pillow Talk? Contracting Collaterizing, and Monetizing Consumer Privacy Through Voice-Captured Personal Data” *Catholic University Journal of Law and Technology*. 2018
6. Bae D., Kim G., Kim J., Park S., Song O. “An Efficient Design of CCMP for Robust Security Network” *School of Electronic Engineering, Chung-Ang University*. International Conference on Information Security and Cryptology. 2006.