Internet of Things (IoT) Device Vulnerabilities and Security

Ben Williams: 957457

May 6, 2020

Abstract

This dissertation explores the Zigbee protocol's security in heterogeneous networks, this is done using a test-bed architecture including a packet sniffer (TI CC2531) and several IoT devices (Amazon Echo and Philips Hue). This project's implementation only evidences exploits using only passive techniques (eavesdropping) but speculates and theorises with what could be achieved with the information collected.

Declaration

This work has not previously been accepted in substance for any degree and is not being currently submitted for any degree.

May 6, 2020

Signed: BWilliams

Statement 1

This dissertation is being submitted in partial fulfilment of the requirements for the degree of a BSc in Computer Science.

May 6, 2020

Signed: BWilliams

Statement 2

This dissertation is the result of my own independent work/investigation, except where otherwise stated. Other sources are specifically acknowledged by clear cross referencing to author, work, and pages using the bibliography/references. I understand that failure to do this amounts to plagiarism and will be considered grounds for failure of this dissertation and the degree examination as a whole.

May 6, 2020

Signed: BWilliams

Statement 3

I hereby give consent for my dissertation to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

May 6, 2020

Signed: BWilliams

Contents

Chapter 1

Introduction

Internet of Things (IoT) is an umbrella term used to refer to any type of device that can communicate over the internet without needing human interaction. This broad definition is reflective of the enormous volume of types of IoT devices currently being sold commercially to a just as large range of end-users. Within the categories of devices there are a massive amount of different physical devices from different companies and organisations. While this diversity in implementation is a marvellous thing for consumer driven markets, it is also the reason why securing IoT networks is so difficult, the relatively newfound pervasiveness of this technology is also why it is so critical to have good security surrounding these devices that end-users can trust. This research aims to analyse and test the current security of certain IoT devices.

The work is based in the context of heterogeneous IoT networks and not only finding vulnerabilities within the devices themselves but also how they communicate with each other in the network. I chose this research topic as I believe simulating a 'normal' consumer's network would result in the most effective output therefore making these results far more potent and useful for the consumer market.

The manufacturing and development of IoT products has become a major market for companies and organisations to such as Google and Amazon, there are currently 26.66 billion IoT devices[4] in the world which is made all the more impressive because of how young the market is as the term itself was only coined in 1999 by Kevin Ashton[5] . All these devices pose a different type of threat to a network, there is a positive exponential correlation between the diversity of devices and the attack surface of a network, the more different devices and ways of connection there are then the more ways there are to attack the network.

This paper will outline the security analysis of two very common IoT devices and the subsequent testing of attack methods based on this analysis on these two devices when set up in a test-bed network.

The dissertation is ordered in the following way:

Chapter 1 Introduction to project with motivations and general research aims.

Chapter 2 Related Works and my contribution.

Chapter 3 Project Outline with specific objectives and description architecture and tools to be used.

Chapter 4 Security Analysis of packet captures and setup of the test-bed network.

Chapter 5 Results of packet analysis and speculation of their implications.

Chapter 6 Evaluation of results against project objectives and review of project characteristics.

1.1 Motivation

The explosion of IoT devices into daily life is what draws me to pursue this project, households around the world are willingly accepting devices such as the Amazon Echo with open arms. The technological expertise level of these end-users has an enormous range, from older users utilizing the voice command features of digital assistants to computer scientists like myself. This range turns IoT device networks into an ocean of opportunity for hackers and a nightmare problem for security experts who are responsible for securing them. The potential damage that a malicious attacker can cause to any IoT network is very interesting an also extremely important when considering defenses that protect against these attacks.

The trade-off between security and performance is a major reason why security in IoT devices is so intriguing since it is a critical juncture that design teams reach in early stages. The methods that companies use to fit security into these LE devices are fascinating although some companies will sacrifice adequate security for performance. The need for a good balance between security and performance is great since having a high-level of both on a LE IoT device is unrealistic because there is simply not enough resources for them.

More and more IoT devices are being developed to automate simple tasks in everyday life which is a beautiful feat and the Smart Homes of the future are fast approaching with some prototypes even being implemented already, with this exchange of control over people's own homes it beckons the question, are the security of these devices up to the task? The need for the security of these devices to be constantly challenged is immense which is why security as a subject should be brought to the forefront in the development of IoT devices.

1.2 Contribution

This dissertation will set itself apart from related works that are similar to this line of thought as it is based on the context of security in heterogeneous IoT networks. Heterogeneous simply meaning that there are multiple devices on the network that have the ability to interact with each other and the hub. This category seperates my research as it is opening up a different type of network, most of the papers discussed in related works test their devices on homogeneous networks. This dissertation will also not provide a final solution or implementation of a product, service or system to deal with fixing the security of IoT devices or networks, it will instead present an analysis of security for two devices and also speculation on what could be accomplished if the security of the network was compromised, and very basic ideas of how to mitigate or reduce these risks. This dissertation will also be focussed on the devices themselves and communication on the network with no focus on cloud endpoint security.

1.3 Project Aims

In short the aims of this project are:

- Create and setup a test-bed network simulating a real consumer network with the ability to intercept packets.

- Analyse packet captures and determine vulnerablities in the network.

- Speculate on implications of the results of packet analysis and passive eavesdropping techniques.

Chapter 2

Related Works

2.1 Research

The security of IoT devices is a critical field and as such many academics have published work that is defined in this context. Alrawi et al [6] created a standard security graph for an IoT network which displayed the attack vectors in the network, with nodes being devices and lines of communication being edges. They concluded that there are four main components of a network that are at risk, devices, companion mobile applications, cloud endpoints and communication edges. Their systematized approach allowed them to categorize and denote 45 devices with these four sections. This paper is similar to what I hope to achieve through my research as security analysis is a major part of my paper.

Fernandes et al [7] analysed the SmartThings platform specifically focussing on the mobile applications that are used in synchronization with the IoT devices, the analysis found that over 55% of SmartApps are over-privileged which is a major security issue facing platforms, this allows leeway for attackers to exploit a network with commands the device does not use. The group also tested exploits of attack vectors using a test-bed network, they achieved four different successful attacks whic included using over-privilege and also utilizing the un-encrypted communication of OAuth tokens over the network. OAuth is an authentication service used in web applications that allows third-party sites to access data from the user without necessarily needing the password. This paper is similar to the work I plan to do since they also test certain attack vectors on a test-bed network.

Miettinen et al [8] created a prototype security system called IoT Sentinel to enforce security on a basic IoT user network. This work is based on the inherently insecure nature of IoT devices, they designed a technique to identify types of devices and categorize them into three different groups based on their security to allow them privileges: strict, restricted and trusted. The system then gives internet access and allows certain communications based on their category. They also demonstrate the scalability of their system using regular consumer products, after this they discuss how to implement traffic flow control on the network to enforce IoT Sentinel. This paper is not similar to my project in the view that this group have created a product or in this case system to improve security on a user network whereas my research will not be implementing a final product or solution to the problem of security.

Geneiatakis et al [9] researched and presented general ideas in their paper ”Privacy and Security Issues in IoT based Smart Home Applications”, this paper gives an example Smart Home network and discusses the general technical information on what is happening in the network, it was extremely useful paper to establish a base knowledge on the current security of IoT devices and in general how these devices work and are established on a network. The paper mentions sensor placements on a network which opened my eyes to just how many different sensors there are to be used or exploited in an adversarial setting. This paper is not similar to my project as while it is based in the security of IoT devices it is only general issues and points without specificity whereas my research will be very specfiic and pertaining to the Amazon Echo and Philips Hue Light.

Al-Fuquha et al [10] published a survey paper on the Internet of Things, this paper provided me with a more advanced knowledge on IoT as a subject and the different types of protocols that these LE (Low Energy) devices use to communicate with specifics about the headers that these protocols such as MQTT use. All of this paper was not useful as it delved deeply into areas like horizontal integration in IoT services which is simply out of the context of my research. This paper is not similar to my research as it is a survey paper and therefore only discusses and provides knowledge of current subjects to give understanding of them.

Qiu et al [11] published a survey paper based on heterogeneous IoT networks and while the whole paper wasn’t useful it did provide a deeper understanding of these networks and security section of the paper was useful for additional thoughts about how having multiple IoT devices on a network can affect and change how security should work on the network. This paper is not similar to mine as once again it is a survey paper with security being a minor part of the overall paper.

Chapter 3

Project Outline

3.1 Objectives

As previously stated the objectives of this dissertation are to:

- Create and setup a test-bed network simulating a real consumer network with the ability to intercept packets.
- Analyse packet captures and determine vulnerablities in the network.
- Speculate on implications of the results of packet analysis and passive eavesdropping techniques.

In more detail the first objective means to build a test-bed that accurately simulates what the average commercial user IoT network would be. This means that common IoT devices should be employed in the network and also that the number of IoT devices themselves should not exceed around 2-3 since IoT devices have a large variety of users with different levels of tech saviness, it is likely that the devices that these consumers use will be the most accessible devices to a consumer of any degree of expertise so the two I have chosen to use are the Philips Hue Light bulbs and the Amazon Echo. The ability to intercept packets is incredibly important as this is the feature that will allow me to to analyse the packets sent and how I will find vulnerabilities in the said network, I have chosen to use a packet sniffer since this is the obvious choice and will buy this device rather than create it myself to limit the scope of this project to be feasible in the time allotted.

The second objective is about analysing the packet captures that have been created from eavesdropping on the test-bed network using the aforementioned packet sniffer. These packet captures through analysis will provide vulnerabilities that could be exploited to attack the network.

The third objective is fairly vague in the sense that the whole objective depends on the results of the second objective and so the form it takes is determined by this. It is however specific in the sense that speculations will be drawn no matter what about the results of the second objective whether they are good or bad.

3.2 Test-bed Archtitecture

The test-bed network of this research will be relatively simple and will consist of a laptop to analyse communication between devices, a packet sniffer device will be connected to this laptop. The laptop will also have a program installed for analysis of packets captured by this device. A smart hub device will be what the user connects and interacts with connected IoT devices through, in this case I have chosen the Philips Hue Bridge as this comes packaged with the Hue light bulbs. This smart hub will send commands using the Zigbee protocols to each device and vice versa, these packets are what will be intercepted using the packet sniffer.

I have chosen to use this architecture for multiple reasons, the first being that it is in my opinion the best setup when considering ease of building and quality of results. The second is that it is the most similar architecture to that found in a common end-user or consumers network, meaning that this architecture will produce a higher quality results which are the most appropriate and most applicable to every commercial user.

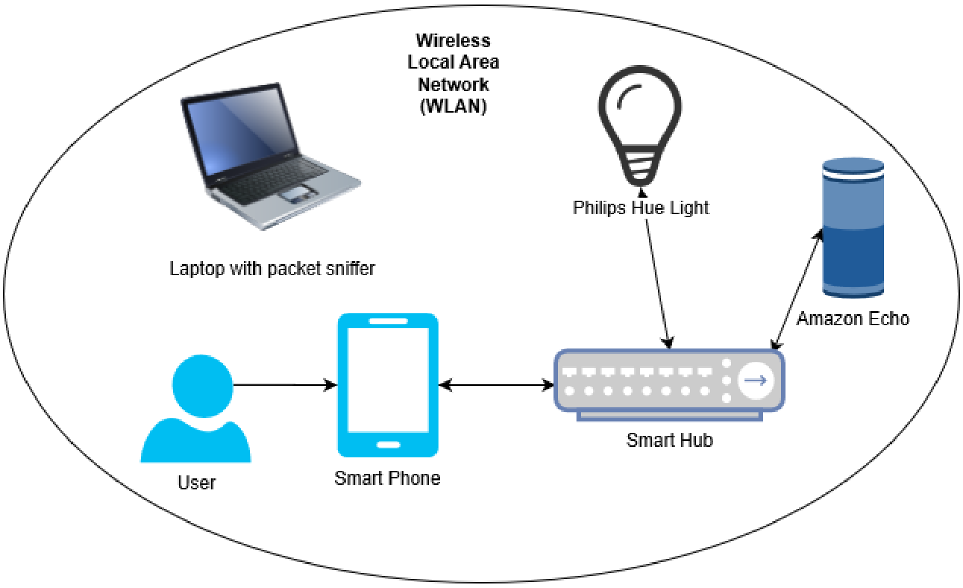


Figure 3.1: Test-bed Architecture

3.3 Tools and Devices

3.3.1 Amazon Echo

The Amazon Echo is a bluetooth speaker integrated with Amazon's personal assistant Alexa which is voice-activated, this allows consumers to have significantly more perks than just speakers, the Alexa can do everything from set alarms to even calling an Uber. This diverse application of features are considered to be the Echo's Unique Selling Point (USP) and therefore it's greatest strength, but this also happens to be it's greatest weakness, it makes the Echo a valuable and viable target when attackers consider a network since not only can it control itself but it can also control other devices on the network if the user has configured them so. The sensor equipment that is implemented in the Echo itself also make it a valuable target with equipment like microphones for eavesdropping purposes.

3.3.2 Philips Hue Light

To be specific this is the Philips Hue Starter Kit E27 IoT Device, it works with the three major personal assistants on the market, these being Amazon Alexa, Apple Homekit and Google Assistant. I have chosen to use this device because of that reason, it enlarges the attack surface area for the network. It is also a very common device type which consumers use and so it will provide a good simulation of the ease of exploit of a common user IoT network. This starter kit comes with a smart hub which is what communicates with the devices over IoT protocols like Zigbee or Z-Wave.

3.3.3 Texas Instruments CC2531 Packet Sniffer [1]

This device manufactured by Texas Instruments is a USB dongle that acts as a packet sniffer, meaning when plugged into a computer and with some configuration it will intercept packets sent over the air in the general vicinity. I specifically chose to use this device over others because it has the ability to intercept Zigbee protocol packets and was alos comparitively the best device on the market when considering reviews and my needs. This device comes with stock firmware flashed onto the chip which means there is less time spent for me on setting up this device, there is a caveat however as the stock firmware only allows for packet sniffing and not packet injection, this for my project however was still suitable.

3.3.4 Wireshark [2]

Wireshark is a free and open source packet analyzer program, the project is hosted by Riverbed Technology. It is a widely used program when analysing a network and is even included in default configurations of Kali Linux (A linux distribution aimed at penetration testing). This program will allow me to read and analyze the Zigbee packets I have captured using their very own Zigbee dissectors, this is a very valuable feature to have as other programs would output the information of the packets in a less clear and concise way. Texas Instruments also have a standard program to capture packets that is paired with their dongles and for a portion of the time of this project this was what I intended on using since it seemed to be the most logical option at the time. This was due to the fact that there was no way to directly use Wireshark with the CC2531 dongle, as my first solution to this.

3.3.5 Texas Instruments Wireshark Packet Converter [3]

The official program created by Texas Instruments to allow converting of packets read by one of their devices to a format that can be read by other programs, with a little shortcut creation can be used in parralel with Wireshark to allow for live packet interception as before it could only do static conversion of the packet capture file. This live feature is very important in my project as it allows me to read and see the packets in real time which aids analysis as I do not have to read over a packet capture file after the fact and try to pinpoint when certain calls or tests were made.

3.3.6 Example Consumer Network

In this section, how a usual user network is set up and the brief details of how it technically works will be explained. The network in 3.2 is relatively simple as it is an average user representation, it consists of a single Smart Hub with three devices connected, these devices only communicate with the Smart Hub as this is the nexus into the IoT network, all traffic flows through it. Hubs will communicate with devices through IoT protocols specially designed for LE devices such as the very popular Zigbee and Z-Wave. The Smart Hub is the bridge between the user and the devices, the user will access device features through the Smart Hub interface once connected, this can happen over either Bluetooth if the user is within the physical geographical boundary of around 10ft, or over the internet through a cloud service. This means the user can control IoT devices in the home from basically anywhere that has a connection to the internet. This feature will not be tested or analysed for security in the scope of this dissertation.

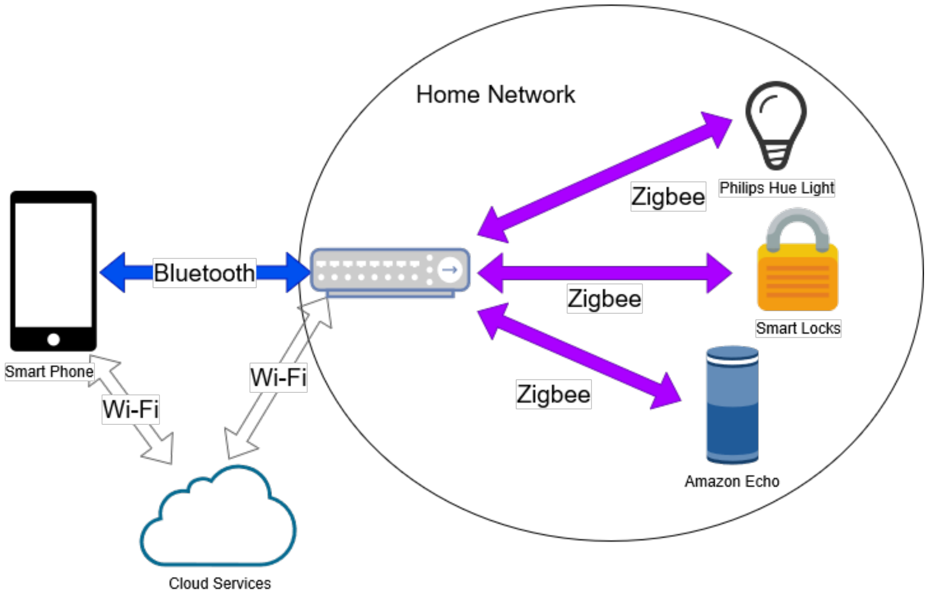


Figure 3.2: Example User Network

Chapter 4

Security Analysis

4.1 Setting up the test environment

This first section of the security analysis will describe the operation of setting up the test environment and also some of the issues faced with their respective solutions that I employed. The test bed is set up as the one described in 3.1 with the specific packet sniffer being the Texas Instruments CC2531 as described in 3.3.3. The Philips Bridge replaces the smart hub.

The first issue I encountered when setting up the testing environment was the configuration of the CC2531 dongle with the laptop used. The device did not register properly with the computer so while it was still connected to the computer and appearing in Device Manager it was not communicating properly with the Texas Instruments SmartRF Packet Sniffer software as this program would not recognize it as a usable device, after some time I figured out the problem was that the drivers had not been correctly installed onto the laptop when the software was installed and I had to manually install these drivers specifically for the CC253x dongles. This fixed the problem and I could now use the dongle and capture packets using the SmartRF program.

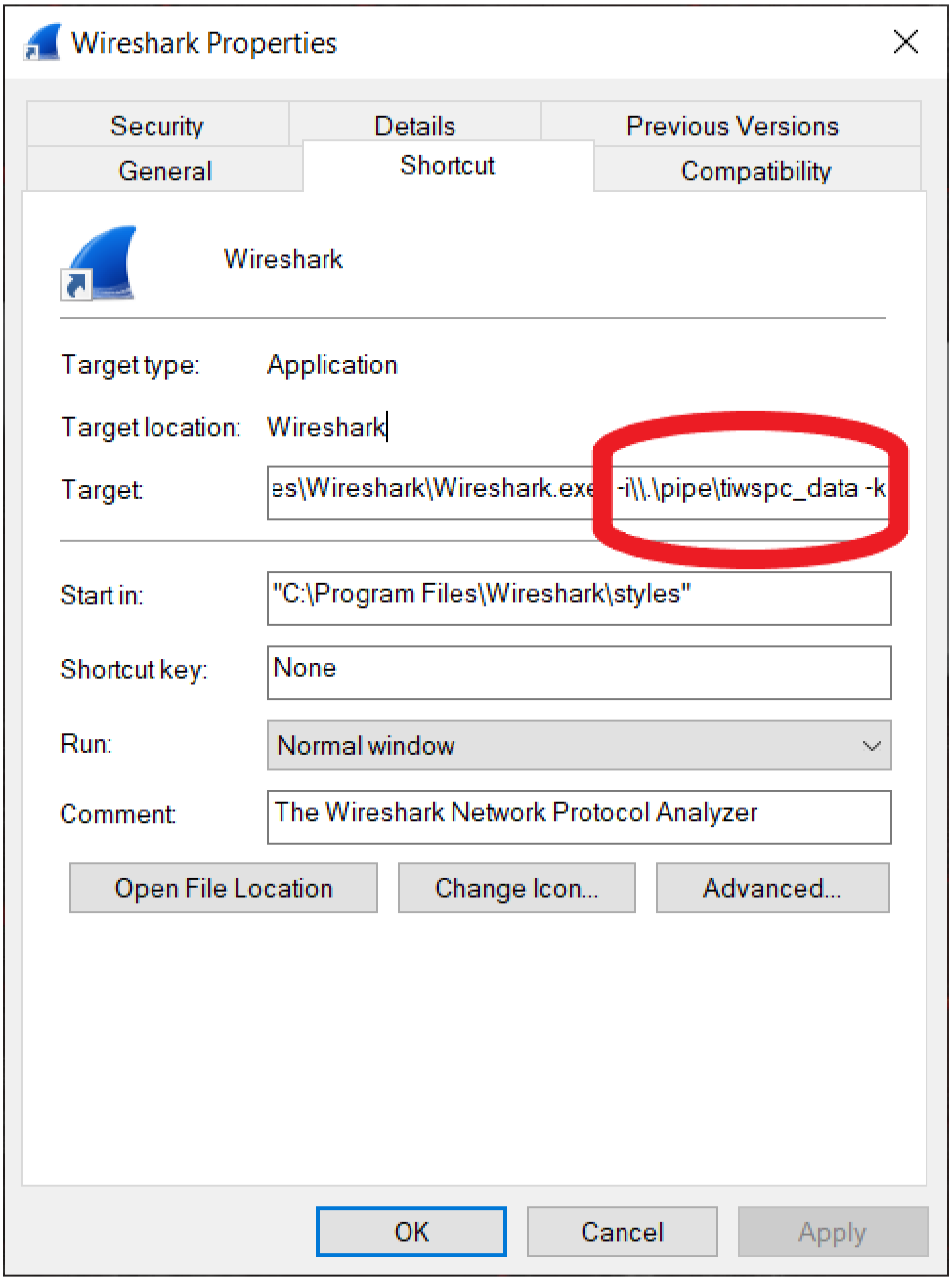
The next problem I encountered was that even though the SmartRF program was made for packet sniffing it was not as clear or easy to use as alternatives such as Wireshark which leads me to the problem that there is no way to use Wireshark directly with the CC2531 dongle that I could find, at this point my solution was to save packet capture files using the SmartRF program and then use a crude java program created by a forum user to convert a .psd packet capture file, which is what the SmartRF program saves, to a .pcap file which is the file extension that Wireshark can save and load from. There were multiple problems with this however the most important being that this program did convert the packets and list them however it shaved off packet numbers and order meaning that a lengthier analysis of the packets would be needed to find the number of each packet and which call they belong to. The second issue was that even if the program was perfect and converted them without mistake it would mean that only static or offline analysis of these packets could be performed where a live analysis of the packets would be preferred. This meant that while this was a solution it was less than ideal and I would need to look for another.

The best solution I came up with to this problem was ironically rather simple, after an exhaustive search through many forums I found a program that will allow the information coming from the device to be piped directly into Wireshark which means live analysis is possible. This program is in fact made by Texas Instruments it is called Wireshark Packet Converter, they however do not advertise this program in any of the convenient places which would have saved me a lot of time, I assume this is because Wireshark is in direct competition with their own Packet Sniffer program. I created a new Wireshark shortcut and appending a string to the end of the target as can be seen in figure 4.1.

Now I had a working packet sniffer with a method of live analysis, the next step was testing that the packet sniffer actually functioned correctly and intercepted Zigbee packets specifically since these are the type of packets I will need to analyse. After an initial setup and pairing of my smartphone to the Philips Hue Bridge I started a packet capture and turned the Hue lights on and off using the Philips Hue app. I knew the packet sniffer functioned properly as I was listening specifically for Zigbee packets and could see that they were streaming in. I had configured the packet converter program to specifically listen on Zigbee channel 11 as this was the channel that the Philips Hue Lights were operating on, I configured this on the Philips Hue App which allows communication on channel 11, 15, 20 and allows changing of certain channels may be busy and so have inteference in certain areas of the house.

4.2 Analysis and configuration of packet captures

The physical interception of Zigbee packets was now possible using the testbed network, the next stage was analysing the packet captures. The problem that I ran into was that even though I was intercepting the packets I could not properly read them since Zigbee uses AES-128 encryption on their packets before transmission to provide confidentiality and stop all unauthorised entities from being able to read the information. AES-128 encryption is a satisfactory standard of encyrption to use, especially in low energy devices such as these IoT devices, it can be broken however with enough time like any other encryption. This is not feasible for this project however since I need to analyse packet captures of the network and each time I assemble the network a new key will be created, fortunately a new solution presented itself



Target type: Application

Target location: Wireshark

Target: es\Wireshark\Wireshark.exe -i\\.\pipe\tiwspc_data -k

Start in: "C:\Program Files\Wireshark\styles"

Shortcut key: None

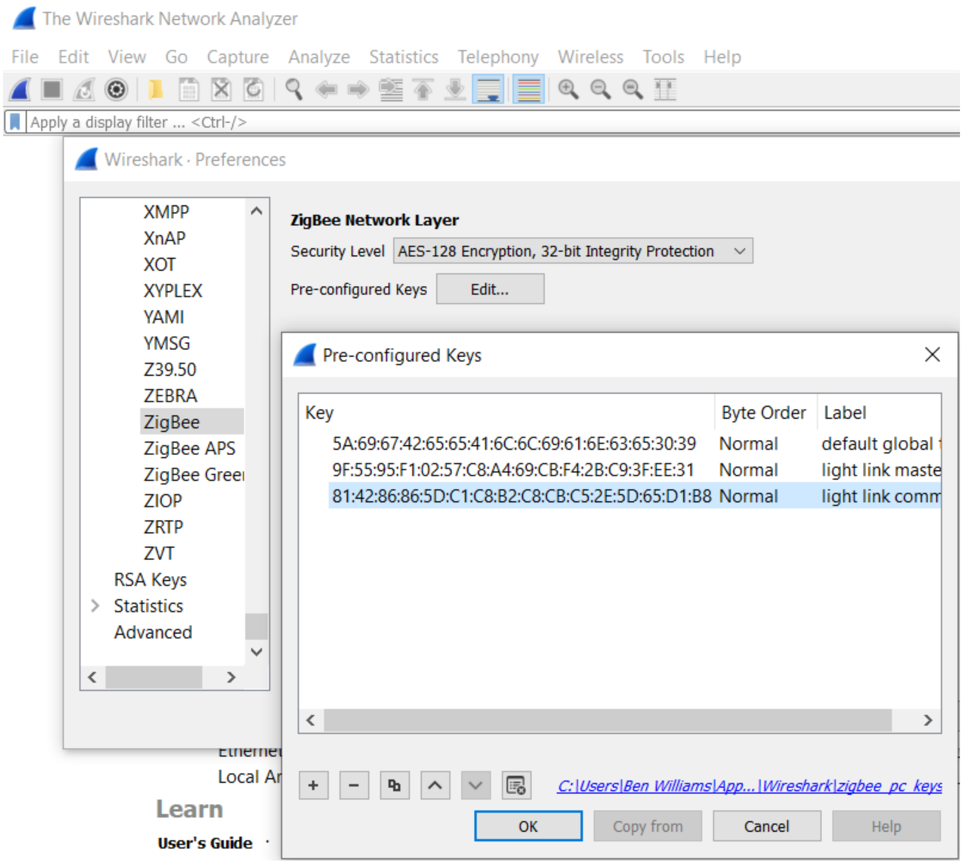
Run: Normal window

Comment: The Wireshark Network Protocol Analyzer

Figure 4.1: Creation of Wireshark Shortcut

through research of papers and articles concerning Wireshark and Zigbee.

Zigbee devices use pre-defined global keys to encrypt each packet using AES-128, these keys however have found their way onto the internet. These keys are the ”default global trust center link key”, this key is used by a trust center to encrypt the network key before it passes it on to a joining device. A trust center is a co-ordinator device which provides the security aspect of a Zigbee network, a co-ordinator device is simply a device on the network that acts as the boss of the network, this includes giving all devices on the network permissions and establishing the network. It will hold all keys needed for security and is responisble for distributing these. The next key is the "light link master key", it allows the Zigbee device to connect to the network and is used during the touchlink phase. or the purposes of this project the last key "light link commissioning key" can be ignored.



ZigBee Network Layer

Security Level: AES-128 Encryption, 32-bit Integrity Protection

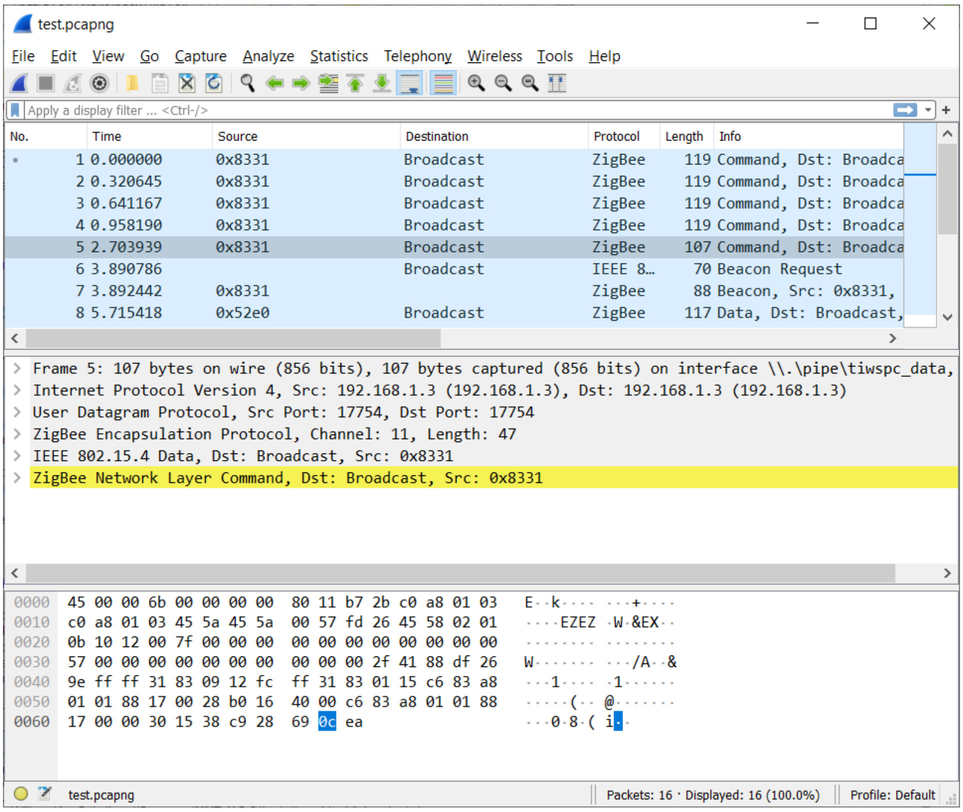
Pre-configured Keys: Edit...

Key	Byte Order	Label
5A:69:67:42:65:65:41:6C:6C:69:61:6E:63:65:30:39	Normal	default global t
9F:55:95:F1:02:57:C8:A4:69:CB:F4:2B:C9:3F:EE:31	Normal	light link maste
81:42:86:86:5D:C1:C8:B2:C8:CB:C5:2E:5D:65:D1:B8	Normal	light link comm

Figure 4.2: Input of Pre-configured Keys

In the test-bed architecture the Philips Hue Bridge acts as the co-ordinator device and therefore the trust center. This means that when light bulbs are first being connected to this bridge then it is sending the network key over Zigbee to the new bulbs, this will become very useful later on.

After finding these keys openly available on the internet I was able to use Wireshark's very useful options and enter these keys in to the pre-configured keys for the Zigbee protocol network layer preference. Now Wireshark's dissectors for Zigbee will automatically use these keys to decrypt the packets in live packet captures or in saved packet capture files. Now I am able to read and see clearly what each packet is communicating as it is using this default global trust center link key to decrpyt all communication on the network.


test.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	0x8331	Broadcast	ZigBee	119	Command, Dst: Broadca
2	0.320645	0x8331	Broadcast	ZigBee	119	Command, Dst: Broadca
3	0.641167	0x8331	Broadcast	ZigBee	119	Command, Dst: Broadca
4	0.958190	0x8331	Broadcast	ZigBee	119	Command, Dst: Broadca
5	2.703939	0x8331	Broadcast	ZigBee	107	Command, Dst: Broadca
6	3.890786		Broadcast	IEEE 8…	70	Beacon Request
7	3.892442	0x8331		ZigBee	88	Beacon, Src: 0x8331,
8	5.715418	0x52e0	Broadcast	ZigBee	117	Data, Dst: Broadcast,

```
> Frame 5: 107 bytes on wire (856 bits), 107 bytes captured (856 bits) on interface \\.\pipe\tiwspc_data,
> Internet Protocol Version 4, Src: 192.168.1.3 (192.168.1.3), Dst: 192.168.1.3 (192.168.1.3)
> User Datagram Protocol, Src Port: 17754, Dst Port: 17754
> ZigBee Encapsulation Protocol, Channel: 11, Length: 47
> IEEE 802.15.4 Data, Dst: Broadcast, Src: 0x8331
> ZigBee Network Layer Command, Dst: Broadcast, Src: 0x8331
```

```
0000  45 00 00 6b 00 00 00 00  80 11 b7 2b c0 a8 01 03   E··k···· ···+····
0010  c0 a8 01 03 45 5a 45 5a  00 57 fd 26 45 58 02 01   ····EZEZ ·W·&EX··
0020  0b 10 12 00 7f 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0030  57 00 00 00 00 00 00 00  00 00 00 2f 41 88 df 26   W······· ···/A··&
0040  9e ff ff 31 83 09 12 fc  ff 31 83 01 15 c6 83 a8   ···1···· ·1······
0050  01 01 88 17 00 28 b0 16  40 00 c6 83 a8 01 01 88   ·····(·· @·······
0060  17 00 00 30 15 38 c9 28  69 0c ea                  ···0·8·( i··
```

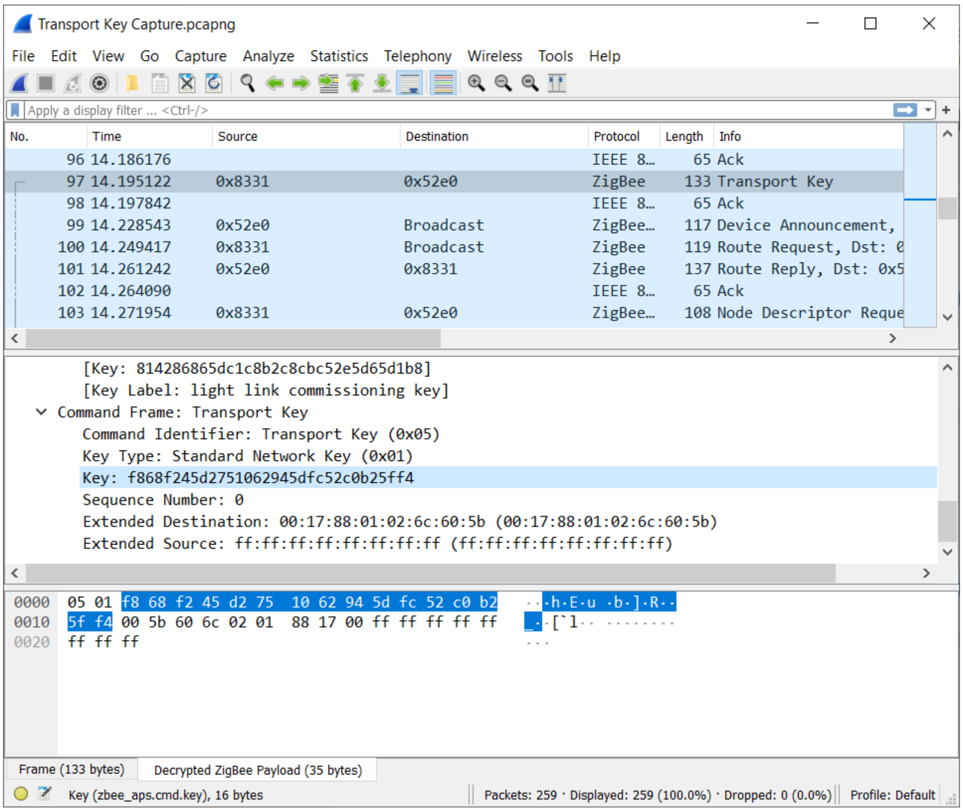
Figure 4.3: Decrypted Wireshark Packet Capture

4.3 Obtaining the network key

With the ability to now read the transmission of Zigbee packets over the testbed network I could now get to work seeing the vulnerabilities of the network and what information a possible attacker could gain through eavesdropping or employing a passive attack on the network. I decided to first try to obtain the network key from the devices, my starting point in theory I have already touched on which is that the network key is passed to all devices when they first try to join the network. If I were to capture the packets that send this information then I would be able to figure what key was being used in the whole network of devices.

I physically unpaired the Hue bulbs using the Philips Hue application on my smartphone and then reconnected them to simulate a regular joining of devices for normal consumers after first purchase. The whole time I was capturing packets using the packet sniffer, I was able to gleam the network key from the packet capture, with this network key being automatically decrypted with the pre-configured keys. This means I can see it in the Wireshark program, by copying this key and placing it into the pre-configured keys as shown earlier and naming it "network key", now all traffic on a heterogeneous Zigbee network can be seen no matter the device because the trust center's security has been compromised. The network key is the base level of security so if it is compromised then the security of the entire network is compromised.

In the results section of the dissertation I will explore what exploits could possibly be achieved with this knowledge.


Transport Key Capture.pcapng
File
Edit
View
Go
Capture
Analyze
Statistics
Telephony
Wireless
Tools
Help
Apply a display filter ... <Ctrl-/>
Frame (133 bytes)
Decrypted ZigBee Payload (35 bytes)
Key (zbee_aps.cmd.key), 16 bytes
Packets: 259 · Displayed: 259 (100.0%) · Dropped: 0 (0.0%)
Profile: Default

No.	Time	Source	Destination	Protocol	Length	Info
96	14.186176			IEEE 8…	65	Ack
97	14.195122	0x8331	0x52e0	ZigBee	133	Transport Key
98	14.197842			IEEE 8…	65	Ack
99	14.228543	0x52e0	Broadcast	ZigBee…	117	Device Announcement,
100	14.249417	0x8331	Broadcast	ZigBee	119	Route Request, Dst: 0
101	14.261242	0x52e0	0x8331	ZigBee	137	Route Reply, Dst: 0x5
102	14.264090			IEEE 8…	65	Ack
103	14.271954	0x8331	0x52e0	ZigBee…	108	Node Descriptor Reque

```
        [Key: 814286865dc1c8b2c8cbc52e5d65d1b8]
        [Key Label: light link commissioning key]
    ⌄ Command Frame: Transport Key
        Command Identifier: Transport Key (0x05)
        Key Type: Standard Network Key (0x01)
        Key: f868f245d2751062945dfc52c0b25ff4
        Sequence Number: 0
        Extended Destination: 00:17:88:01:02:6c:60:5b (00:17:88:01:02:6c:60:5b)
        Extended Source: ff:ff:ff:ff:ff:ff:ff:ff (ff:ff:ff:ff:ff:ff:ff:ff)
```

```
0000  05 01 f8 68 f2 45 d2 75  10 62 94 5d fc 52 c0 b2   ···h·E·u ·b·]·R··
0010  5f f4 00 5b 60 6c 02 01  88 17 00 ff ff ff ff ff   _··[`l·· ········
0020  ff ff ff                                           ···
```

Figure 4.4: Network key captured and automatically decrypted

Chapter 5

Results and implications

This section of the dissertation will be restating results found through investigation and then speculating on these results and theorising based on similar research on what could be done by a malicious attacker with the information found.

The biggest result of this dissertation is the eavesdropping of the network key in section 4.3. The network key is used to encrypt all traffic that is sent on the network by all devices including trust centers or co-ordinators, if a malicious attacker had this key then similar to what I have achieved with the help of a packet sniffer device would be able to eavesdrop and see all traffic on the Zigbee network. This poses an incredible threat as privacy is breached, the attacker is able to see all commands sent in the network, this may not seem like a massive risk as they might not be able to do anything witht his information I assure you they can.

By simply packet capturing on Wireshark the attacker can see all sources and destinations meaning by simply performing a set function they are able to determine how many Zigbee devices are on this network. Again this may not seem like a huge risk but with the packet capture they can also see the commands as previously stated, for different types of devices these commands will be different and therefore with a little research the attacker can determine which devices are of which type. In addition to all this information they could also determine which devices on the network are trust centers or co-ordinators. Experienced attackers could leverage this information in an impersonation attack for example, this attack would involve changing the source field of a packet to the co-ordinator and injecting this packet with a malicious command, a good example of this relating to my test network would be the impersonation of the Hue bridge and a command being sent to turn the light on.

There are many implications if an attacker has the ability to eavesdrop on a Zigbee network, real world implications could be very serious. For this I will create a scenario, a regular user who has rigged up Hue lights throughout their house to be controlled by the Hue app on their smartphone. They have used the "rooms" and "scene" features that are present on the app to group lights, set up routines and turn lights on and off based on proximity with the use of location data of their phone. Now looking at the attackers side of the scenario, the attacker has to do three things to eavesdrop on the network is to purchase a packet sniffer which is fairly easy, the same as buying anything online, be able to have a close physical proximity to the target network and capture the network key the same way I have. Zigbee has a range of 10-20 meters meaning in most consumer households such as terrace houses then a packet sniffer can be placed just outside the household and still recieve the packets. Granted that to gather a lot of the information a long time window is needed, this could however be sorted either by a portable capture storage medium or a relay of some kind to send the information to another computer. The attacker could either capture the network key when the user is first setting up the Zigbee devices or somehow simulate the joining of a device to the network, now the attacker can view all traffic on the network. If they were to capture a two day period for example, they could see the timestamps of the commands, using the "room" feature of the app means that when a user presses the button to turn on that room then a command is sent to each light attached to that "room" to turn on. The attacker can see the timestamps of commands and if there were three lights turned on at a certain time then this would be one room, using this thinking they could create an algorithm to search for commands sent at the same time and then determine all rooms in the user's household. Other pieces of information could be gleamed from these timestamps, for example if all lights are turned off at a certain time in the night then a sleeping pattern could be formed. This information could be very useful to malicious attackers and the scary part of this fact is that an attacker could map out the rooms and structure of a user's house simply because they have Zigbee lights, which out of all Zigbee enabled devices would be considered to have the least commands and permissions, simply turn on and off commands.

This issue becomes even more dangerous if the user were to have the Amazon Echo or Smart locks, every different type of device will give attackers new pieces of information that could be maliciously leveraged against the users. Considering that I have not even written about the more dangerous half of packet methods, this being packet injection, just emphasises why the need for security is absolutely critical in IoT devices.

I will now theorise and speculate on different types of attacks that could be used with the information gathered through packet sniffing, these attacks will be focused on attacks that could be performed with packet injection as if I were to continue this project in the future, packet injection would be avenue I would pursue.

The first attack vector to discuss I have already mentioned in this section, it is impersonation attacks in a Zigbee network. The general basis of this attack is an attacker masquerades as an official entity and either sends or recieves information that is meant or the entity it is pretending to be, the most basic example of this would be an attacker pretending to be a line manager in an office through emails and would then recieve all the emails sent to the line manager by their subordinates. Adapting this attack for packet injection would be as previously described, the source of the packet would be faked and changed to an official device on the network so when the destination device recieves the packet they think it is from the official device but it contains a malicious payload. The effect of this attack is multiplied by how many permissions the official entity that the attacker is pretending to be has and also at what point in the "food chain" the device is. By the "food chain" I mean whether the device is an end device such as a Hue light bulb or a co-ordinator like the Hue Bridge or even higher such as a trust center since trust centers are charged with ensuring the security of the whole network. An example of an impersonation attack could be taking advantage of an Amazon Echo which has many permissions in an IoT network and using it to affect other devices in a network such as turning on lights or affecting other IoT devices that have been paired with the Amazon Echo.

I believe that packet injection is significantly more dangerous than packet sniffing since it can physically affect user devices on a network rather than gleaming information about the network, however I do recognize that both are dangerous. The problem with IoT devices is that these devices are often low energy and resource deprived meaning that "beefy" security that uses a lot of those resources is not realistic. This gap in security that occurs when configuring a joining device for the first time on a network is recognized by the creators of Zigbee as referenced in the MIT Zigbee paper [12] and they consider this to not be a part of their security since before a device joins a Zigbee network it is not considered to be secured and at the point where the network key is transported to the device is the point where the device is considered to be a part of the security of the Zigbee network.

Chapter 6

Evaluation

6.1 Objectives

In this section I will restate the three project objectives and evaluate whether I have met them. The objectives for this project were:

- Create and setup a test-bed network simulating a real consumer network with the ability to intercept packets.
- Analyse packet captures and determine vulnerablities in the network.
- Speculate on implications of the results of packet analysis and passive eavesdropping techniques.

The first objective was to create and setup a test-bed network, this involved gathering the proper devices that a consumer would use, these being the Amazon Echo and Philips Hue Lights, the next part of the objective was having the ability to intercept packets and essentially use packet sniffing. This was achieved through the purchasing of the TI CC2531 dongle after thorough research of review and specification, since I was dealing with devices that employed Zigbee I had to make sure the packet sniffer was able to intercept Zigbee packets. The CC2531 is able to intercept packets on all Zigbee channels and therefore was the most suited tool to complete the task. There was a third part of this objective which was partly hidden, this being simulating what the normal user would be doing on the network which is fairly easy to achieve as it is just using the devices as they were intended and not doing anything too technically difficult for the average user, this includes not configuring security on the devices as this coould be considered as technical. Overall I have achieved this objective as using the packet sniffer I could intercept and form a static packet capture of the network traffic, I even went beyond this as I was not satisfied by just the static packet capture and as previously described found a way to pipe the data directly into Wireshark and have a live packet capture in Wireshark of the Zigbee network traffic alone.

The second objective being to analyse the packet captures was achieved as I was able to decrypt all packet captures using the pre-configured keys that all Zigbee devices use, this allowed me to find the packet which contained the network key and then add this to the pre-configured keys feature in Wireshark for the Zigbee protocol, this is what enabled me to analyse all packets as they were all decrypted using this network key. The second part of the objective was to determine vulnerabilities in the network, I believe I have achieved this as this window where the network key is given to devices joining the network for the first time is in fact a vulnerability, it was previosuly described in the MIT Zigbee Paper[12] and has been succesfully recreated in this project. Overall I believe this objective has been achieved and evidenced in this project thoroughly.

The third objective was based on the results from the first two objectives after completion, it was achieved through research and speculation, I believe I have achieved the objective as I have discussed what could be achieved through the knowledge of the network key and the ability to read all traffic on a network once decrypted. I have also discussed some attacks that could be performed such as the impersonation attack or the replay attack.

Overall, for the project I believe I have achieved these three objectives and it has resulted in a substantial piece of work that has benefited from my previous knowledge especially from CSC318 Cryptography and IT-Security, it as also benefited from the Computer Science avenue that I have learned over the past three years at university as this was not a typical software engineering project that requires hundreds of lines of code as this discipline has been taught in various other modules but instead this project required experimentation and better understanding of the theory behind topics.

6.2 Risks and mitigation

A risk that occured during the project that I definitely did not see coming was the Coronavirus pandemic, this pandemic obviously affected the quantity and quality of the work done in the project, in many ways such as the time taken away in doing replacement work in other university modules and time I couldn't access some of the equipment such as the Amazon Echo. I do not think this is a detriment to myself though as I highly doubt that anyone could have foreseen this global pandemic coming and the effects it would have on just about everything.

A risk that also affected the project was that the amount of time it would take to complete this scale of project was an estimate, this was affected by many issues such as the pandemic but overall this risk did not occur as the project was still finished in time, however if these didn't occur maybe extra objectives could have been met or just the original objectives better met.

Communication between myself and my supervisor was also a risk which was emphasised by the pandemic however all e-mails and other communications were all met and replied to in sufficient time, as such no mitigation methods were needed or employed since communication in the present is easily sufficient.

The final risk that was most likely to happen was the damage or event that would cause my laptop to become unusable, this risk did not occur within the time frame of the project but this does not mean that the mitigation strategies were not employed to deal with it if it had occured. A private GitHub repository was created and updated every two weeks with the files on my laptop. It was created as a private repository as to not break the rules of academic misconduct as only I can access it, all modified files were staged then comitted and finally pushed to the repository.

6.3 Time Management

Overall the project has been accomplished on time but the time management could have been a lot better. The first issue with time management in the project came with the determination and ordering of the packet sniffer device I would be using for the project, this was different to the other equipment I was using since the rest was already on site in the unviersity, the packet sniffer I ordered (TI CC2531) took up about a week to decide this was the one I should purchase through comparison of reviews. Then it took two weeks for me to able to use since I did not foresee the time it would take for the university to confirm the order and then for it to arrive. This did affect the project quite a bit since at the point when the device did arrive 3-4 courseworks were given out and/or due in. This delayed the actual testing and implementation of the project and limited what I could accomplish.

The second and final issue was the time impact that the global pandemic of COVID-19 had on the project which was significant because of the knock-on effect from all the other university labs and courseworks, other priorities were re-arranged due to this since replacement courseworks were created for labs and other work that needed to be done in person. This particularly affected me since in CSC368 Embedded Systems to replace labs where we working with physical robots a replacement coursework was created, however it was created in the last week before the easter break and as such affected the time I had to complete the project and write this very dissertation.

In general, time management went well over the whole project, it was only these two issues which were in the latter half of the project which affected it. The first issue I did not foresee but in fairness it could have been foreseen that ordering and delivery of specialist equipment could take longer than expected, however when undertaking similar projects in the future I will know this and can take it into account when planning the schedule of a project. The second issue I also did not foresee was the COVID-19 pandemic but in fairness I don't think anyone could have foreseen this sort of issue and on such a large scale however I still believe I can learn from this issue for future projects, for example in future projects I will strive to keep access to necessary equipment, I will also schedule tests and objectives in batches and perform them as such so even if some time is taken away after a batch is complete it will not affect the project as much as it would otherwise. This is a much more efficient way of handling a project rather than a continuous implementation, it is also seen in agile software development strategies.

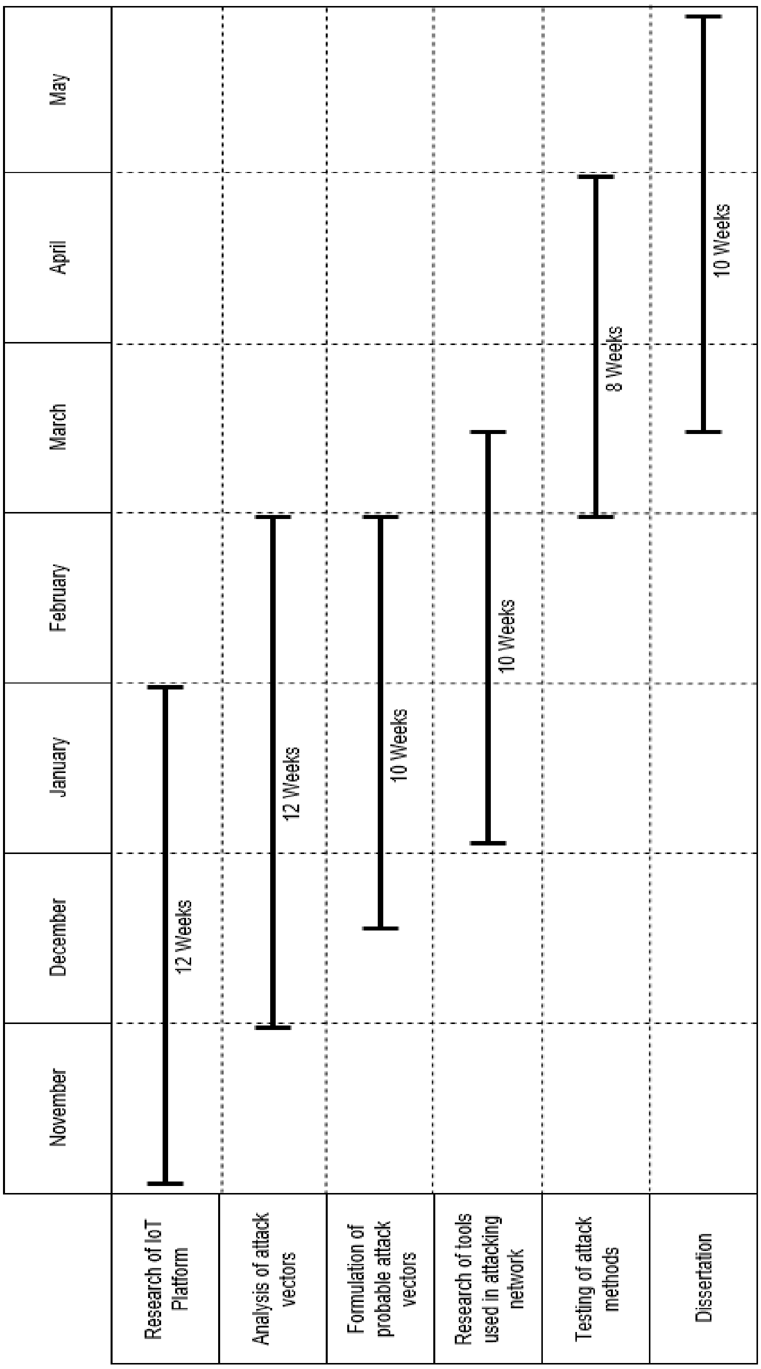


Figure 6.1: Gantt chart from initial document

Unfortunately when starting this project and at the time of submitting the initial document, the project has changed shape and while the project was completed in time, this initial Gantt chart was naive and very differently scheduled to what the project actually looked like. The parts of the project that looked similar to what was included in the Gantt chart were the research and dissertation sections, the main part of the research section was achieved in the time frame of 12 weeks from November to the end of January however I was naive about thinking that all research would be accomplished in this timeframe since during the course of a project its flow changes and new problems arise which need research to develop a solution and new areas open up which need to be explored to determine what they mean about the project. Research was continuous until close to the end of the project around the middle of April.

The objectives themselves had changed since the writing of the initial document to be more specific and fiiting to the end result for the project that I wanted to achieve and so some of the sections in the Gantt chart are not applicable, aforementioned delays of access to equipment also extended sections of the project such as the gathering of the data and analysis of it, obviously delays to speculation of results were necessary because of this. What I have learned from this however is in future to extend and pad certain areas to account for unforeseen delays to produce a better schedule which will accomodate these problems and the outcome will be a more realistic representation of what the final schedule will turn out to be.

6.4 Future

In this project I was not able to use packet injection to test what could be achieved through the use of the gathered keys and information from the packet analysis and packet sniffing, if I were to continue this project in the future I would definitely explore packet injection to test attack vectors and methods such as replay attacks on the Zigbee protocol. I have researched this somewhat and replay attacks in general do not work since the protocol and devices take into account packet numbers but using firmware in combination with software such as Killerbee, a packet injection and sniffing open source project, would allow me to fake the packet numbers. I was not able to explore packet injection due to time constraints I mentioned previously about ordering the packet sniffer device, to be able to use packet injection I would need to first find suitable firmware like Killerbee and flash it onto the board of the CC2531 Dongle through a different device called a CC Debugger, this whole process would have taken around a month extra to complete since I would first have to order this new device considering the same time scale as the CC2531 Dongle of two weeks to arrive, then a week or so actually completing the process since I would have to learn new technology of connecting the pins and such, then an extra few days to deal with unforeseen problems of learning a new technology.

6.5 Acknowledgements

I would like to acknowledge my supervisor Dr. Pardeep Kumar for consistent communication over the course of the project aswell as giving me the support needed, as well as proviiding knowledge of protocols and background information surrounding AES encryption in the CSC318 Cryptography and IT-Security module in the first semester of 3rd year.

Bibliography

[1] Texas Instruments. CC2531 Zigbee and IEEE 802.15.4 wireless MCU with up to 256kb Flash and 8kB RAM. http://www.ti.com/product/CC2531, 2020. Accessed: 2020-04-14.

[2] Riverbed Technologies. Wireshark Go Deep. https://www.wireshark.org/, 2020. Accessed: 2020-04-14.

[3] Texas Instruments. PACKET-SNIFFER SmartRF Protocol Packet Sniffer. http://www.ti.com/tool/PACKET-SNIFFER, 2020. Accessed: 2020-04-14.

[4] A. Bera. 80 IoT Statistics for 2019. https://safeatlast.co/blog/iot-statistics/, 2019. Accessed: 2020-01-22.

[5] K. Claveria. Meet Kevin Ashton, the visionary technologist who named the Internet of Things. https://www.visioncritical.com/blog/kevin-ashton-internet-of-things, 2019. Accessed: 2020-01-22.

[6] O. Alrawi & C. Lever & M. Antonakakis & F. Monrose. SoK: Security Evaluation of Home-Based IoT Deployments. In *Proceedings of the 2019 IEEE Symposium on Security and Privacy*. IEEE Computer Society, 2019. Retrieved from https://alrawi.github.io/static/papers/alrawi_sok_sp19.pdf.

[7] E. Fernandes & J. Jung & A. Prakash. Security Analysis of Emerging Smart Home Applications. In *Proceedings of the 2016 IEEE Symposium on Security and Privacy*. IEEE Computer Society, 2016. Retrieved from http://iotsecurity.eecs.umich.edu/img/Fernandes_SmartThingsSP16.pdf.

[8] M. Miettinen & S. Marchal & I. Hafeez & A.R. Sadeghi & N. Asokan & S. Tarkoma. Iot Sentinel: Automated Device-Type Identification for Security Enforcement in IoT. In *IEEE 37th Conference on Distributed Computing Systems*. IEEE Computer Society, 2017. Retrieved from https://arxiv.org/pdf/1611.04880.pdf.

[9] D. Geneiatakis & I. Kounelis & R. Neisse & I. Nai-Fovino & G. Steri & G. Baldini. Security and Privacy Issues for an IoT based Smart Home. In *IEEE 37th Conference on Distributed Computing Systems*. MIPRO, 2017. Retrieved from `https://www.ijert.org/research/privacy-and-security-issues-in-iot-based-smart-home-applications-IJERTCONV6IS15010.pdf`.

[10] A. Al-Fuqaha & M. Guizani & M. Mohammadi & M. Aledhari & M. Ayyash. Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications. *IEEE Communication Surveys & Tutorials*, 17, 2015. Retrieved from `https://ieeexplore.ieee.org/document/7123563`.

[11] T. Qiu & N. Chen & K. Li & M. Atiquzzaman & W. Zhao. How Can Heterogenous Internet of Things Build Our Future. *IEEE Communication Surveys & Tutorials*, 20, 2018. Retrieved from `https://www.researchgate.net/publication/323059293_How_Can_Heterogeneous_Internet_of_Things_Build_our_Future_A_Survey`.

[12] X. Fan & F. Susan & W. Long & S. Li. Security Analysis of Zigbee. In *MIT Research Paper*. MIT, 2017. Retrieved from `https://courses.csail.mit.edu/6.857/2017/project/17.pdf`.