Graduate thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Information Security and Communication Technology

Andersen, Markus

Identification, Location Tracking and Eavesdropping on Individuals by Wireless Local Area Communications

Graduate thesis in MTKOM

Supervisor: Mjølsnes, Stig Frode

June 2019

NTNU
Norwegian University of
Science and Technology

Problem Description

Title: Identification, Location Tracking and Eavesdropping on Individuals by Wireless Local Area Communications
Student: Markus Andersen

Personal mobile terminals, such as smartphones, laptops, and tablets, are equipped with Wi-Fi and Bluetooth as standard wireless communication means. This master's thesis will investigate the technical possibilities for clandestine identification, location tracking and eavesdropping on individuals carrying such communication terminals, by exploiting functionalities available in the wireless communication protocols and their implementations.

The thesis work should select the best published methods for clandestine location tracking, and then establish the feasibility and limitations of those methods by setting up and carrying out suitable experiments and trials for this purpose. Furthermore, the candidate may be able to construct new methods for location tracking and eavesdropping as the investigation progresses, and propose technical and procedural means of avoiding such privacy attacks.

Responsible professor: Stig Frode Mjølsnes
Supervisor: Vegard Antonsen
Supervisor: Torjus Bryne Retterstøl

Abstract

Almost all user equipment with wireless capabilities are created with hardware chips in-bedded with a MAC-address. A MAC-address is a unique ID, used in layer two of the OSI model [1], which is used to separate user equipment in Local Area Network (LAN). A MAC-address is defined as personal data when it is collected through Wi-Fi and Bluetooth tracking [2].

This master's thesis covers clandestine and active location tracking and eavesdropping of mobile user equipment through Wi-Fi and Bluetooth. The thesis establishes the feasibility and limitations of best published methods for clandestine and active location tracking and eavesdropping. This is established by setting up and carrying out suitable experiments for this purpose.

This thesis provides results based on the experiments carried out for both their feasibility and their limitations, which proved that both clandestine and active location tracking is very simple and highly feasible. Eavesdropping is much more complicated, which is backed by its limitations and lower feasibility, but still possible.

This thesis contributes to improve privacy of the protocols Wi-Fi and Bluetooth with proposed technical and procedural means of avoiding such privacy attacks. Amongst other proposed improvements is the randomization of the full MAC-address of six bytes for every connection and the implementation of public and private certificates.

Sammendrag

Nesten alt brukerutstyr med trådløs funksjonalitet har maskinvare som har en MAC-adresse innebygd. En MAC-adresse er en unik ID, i lag to av OSI-modellen [1], for å separere brukerutstyr i lokale nettverk (LAN). En MAC-adresse er definert som personlig data dersom den er innsamlet gjennom Wi-Fi- eller Blåtann-sporing [2].

Denne masteroppgaven dekker passiv og aktiv sporing og avlytting av mobilt brukerutstyr gjennom Wi-Fi og Blåtann. Oppgaven etablerer gjennomførbarhet og begrensninger tilknyttet de beste publiserte metodene for passiv og aktiv sporing og avlytting. Dette er etablert gjennom oppsett og utføring av passende eksperiment for dette formålet.

Oppgaven gir resultater basert på gjennomførbarhet og begrensninger for eksperimentene utført, som beviste at både passiv og aktiv sporing er veldig enkelt med høy grad av gjennomførbarhet. Avlytting er mye mer komplisert, som er støttet opp av sine begrensninger og lave grad av gjennomførbarhet, men er fortsatt mulig.

Masteroppgaven bidrar til å øke personvernet knyttet til protokollene Wi-Fi og Blåtann med foreslåtte tekniske og prosedyriske midler for å unngå disse personvernsangrep. Blant foreslåtte forbedringer er randomisering av hele MAC-adressen på seks byte for hver tilkobling og implementering av offentlige og private sertifikater.

Preface

This is the final report of the work with the Master's thesis in Information Security in the 10th semester of my Master of Science degree in Communication Technology at the Norwegian University of Science and Technology.

Firstly I would like to thank my mother for lending me her home office, and my father for bringing me a countless number cups of coffee.

Secondly I would like to thank my responsible Professor Stig Frode Mjølsnes for great feedback, and my supervisors Vegard Antonsen and Torjus Bryne Retterstøl for much valued guidance, feedback and discussions. Without their help this thesis would not include the same amount of research or methods used during testing, and would not achieve the same results.

Lastly I would like to thank the Norwegian National Security Authority (NSM) for the their interest in this thesis, giving me an office, a supervisor and paying for all experimental equipment.

Contents

List of Tables **xiii**

List of Figures **xvi**

List of Acronyms **xvii**

1 Introduction **1**

1.1 Motivation 1

1.2 Scope and Objectives 2

1.2.1 Scope 2

1.2.2 Objectives 2

1.3 Work Method 3

1.3.1 Phases 3

1.3.2 Cyber Kill Chain 3

1.4 Methodology 4

1.5 Contributions 5

1.6 Prerequisites and Limitations 5

1.7 Related Work 5

1.7.1 Improvements to Androids Wireless Fidelity (Wi-Fi) 5

1.7.2 Improvements to iOS' Wi-Fi 5

1.7.3 SINTEF: Eavesdropping on Hearing Aids 6

1.7.4 How talkative is your mobile device? 7

1.7.5 On Tracking the Physicality of Wi-Fi: A Subspace Approach 7

1.8 Discussion 7

1.9 Outline 8

2 IEEE 802.11 Wireless **9**

2.1 Overview 9

2.2 IEEE 802.11 9

2.2.1 Probe Request 9

2.2.2 Beacon And Probe Response Frame 10

2.2.3 IPv6 12
2.2.4 WPA3-Personal 12
2.3 Wi-Fi Probe Sniffing 12
2.4 Access Point Replicating 13

3 Bluetooth **15**
3.1 Bluetooth Classic 15
3.1.1 Bluetooth Connection Handshake 15
3.2 Bluetooth Low Energy 16
3.2.1 Security Modes 17
3.3 Bluetooth Media Access Control address (MAC-address) 18

4 Experiment **21**
4.1 Experimental Setup 21
4.1.1 Raspberry Pi 22
4.1.2 Alfa USB Wireless Adapter Setup 23
4.1.3 Hak5 WiFi Pineapple Setup 24
4.1.4 Hak5 WiFi Pineapple Windows 7 Setup 26
4.1.5 Kismet Setup 26
4.1.6 Ubertooth Setup 28
4.1.7 Bettercap Setup 31
4.1.8 Gattack Setup 32
4.1.9 Crackle Setup 34
4.2 Other Available Software and Hardware 34
4.2.1 Software Defined Radio 35
4.2.2 Bluefruit LE Sniffer 35
4.2.3 nRF Sniffer 35
4.2.4 GNU Radio 35
4.2.5 Micro:Bit, Btlejuice and Btlejack 35
4.3 Method of Choice Obtaining MAC-addresses and Man-in-The-Middle (MiTM)-attacks 35
4.3.1 Passive Wi-Fi Sniffing 36
4.3.2 Passive Bluetooth Sniffing 36
4.3.3 Active Wi-Fi Sniffing and MiTM-attack 36
4.3.4 Active Bluetooth Sniffing and MiTM-attack 37
4.4 Discussion 37

5 Results **39**
5.1 Passive Location Tracking 39
5.1.1 Passive Wi-Fi Sniffing 39
5.1.2 Passive Bluetooth Sniffing 42
5.2 Active Location Tracking 44
5.2.1 Active Wi-Fi Sniffing 44
5.2.2 Active Bluetooth Sniffing 45
5.3 Eavesdropping 46
5.3.1 Wireless MiTM Attack 46

5.3.2 Bluetooth MiTM Attack 47
5.4 Correlation Between Wi-Fi and Bluetooth 47
5.4.1 Correlation on Android 8.0.0 47
5.4.2 Correlation on iOS 12.3.1 48
5.5 Discussion 49

6 Countermeasures 53
6.1 Anonymous Probe Requests 53
6.2 Disabling Directed Probe Requests 54
6.3 Randomization of MAC-address 54
6.4 Certificates and Public/Private-Key Encryption 54
6.5 Discussion 57

7 Discussion 59

8 Conclusion 63
8.1 Future work 64

Bibliography 65

List of Tables

1.1 Objectives for this thesis. 3
1.2 Cyber Kill Chain. 4

3.1 Bluetooth Low Energy Security Levels. 17
3.2 Bluetooth Low Energy Security Modes. 18

4.1 Features of Hak5's Pineapple. 25
4.2 Features of the Bettercap software. 31
4.3 Gattack modes of operation. 33
4.4 Crackle modes of operation. 34
4.5 Features of the Hak5's Pinapple Nano. 37

5.1 Comparison between the different attack methods. 51

List of Figures

1.1 German Police released photo of wanted DHL parcel bomber with MAC address. 2

2.1 Probe request frame with associated sizes. 10
2.2 Probe broadcast request as it would be viewed in Wireshark. 10
2.3 Probe response frame with associated sizes. 11
2.4 Probe response as it would be viewed in Wireshark. 11
2.5 Local IPv6 inherits the MAC-address of the User Equipment (UE). . 12
2.6 Screenshot of *wigle.net* (Accessed 2019-05-31). 13

3.1 Bluetooth connection handshake. 16
3.2 The different parts of the MAC-address [3] (Accessed 2019-05-19). . 18

4.1 Hardware used in the experiment. 22
4.2 Raspberry Pi 23
4.3 Alfa Wireless Adapter (left) and Hak5 WiFi Pineapple Nano (right). 24
4.4 Devices and printers, Properties. 26
4.5 Enable driver. 26
4.6 Kismet running from terminal. 27
4.7 Kismet Data Sources shown in web browser. 28
4.8 Bluetooth wireless adapter (left) and Ubertooth (right). 29
4.9 Ubertooth-rx options (*ubertooth-rx -h*). 30
4.10 Bettercap running with current Events [4]. 33

5.1 Probe request including its real MAC-address. 40
5.2 Kismet showing nearby devices including their MAC-address. 41
5.3 Pineapple recon function showing nearby devices including their MAC-address. 41
5.4 Pineapple recon function showing Access Point (AP)s and its currently connected devices. 42

5.5 Ubertooth-scan function calculating the Adaptive Frequency-Hopping (AFH). 43
5.6 Ubertooth passive sniffing for Lower Address Part (LAP) and Upper Address Part (UAP) of MAC-address. 44
5.7 Pineapple using Recon function. 45
5.8 Bluetoothctl using scan function. 46
5.9 Correlation between Wi-Fi and bluetooth on Android version 8.0.0. . . 48
5.10 Correlation between Wi-Fi and bluetooth on iOS version 12.3.1. . . . 49

6.1 Probe broadcast request as it would be viewed in Wireshark. 53
6.2 Public 802.11-certificate. 55
6.3 Private 802.11-certificate. 56

List of Acronyms

AES-CMAC	Advanced Encryption Standard - Cipher-based Message Authentication Code
AES	Advanced Encryption Standard
AFH	Adaptive Frequency-Hopping
AP	Access Point
ARP-DeAuth	Address Resolution Protocol Deauthentication
BLE	Bluetooth Low Energy
BSSID	Basic Service Set Identifier
CA	Certificate Authority
CKC	Cyber Kill Chain
CSI	Channel State Information
DES	Data Encryption Standard
ECDH	Elliptic-Curve Diffie-Hellman
ENISA	European Union Agency for Network and Information Security
EU	European Union
FHSS	Frequency Hopping Spread Spectrum
GDPR	General Data Protection Regulation
GHz	Gigahertz
GNU	GNU's Not Unix
GPS	Global Positioning System
GUI	Graphical User Interface
HTTPS	Hypertext Transfer Protocol Secure
HTTP	Hypertext Transfer Protocol
IMSI	International Mobile Subscriber Identity
IMEI	International Mobile Equipment Identity

IOT	Internet-Of-Things
ISP	Internet Service Provider
LAN	Local Area Network
LAP	Lower Address Part
LTK	Long Term Key
MAC-address	Media Access Control address
MiTM	Man-in-The-Middle
NAP	Non-significant Address Part
NPM	Node Package Manager
NSM	Norwegian National Security Authority
NTNU	Norges Teknisk-Naturvitenskapelige Universitet
OSI BRM	Open Systems Interconnection Basic Reference Model
OS	Operating System
OUI	Organizationally Unique Identifier
POTS	Public-of-the-Shelf
PSK	Preshared Key
PnP	Plug-and-Play
RSA	Rivest–Shamir–Adleman
SAE	Simultaneous Authentication of Equals
SDN	Software Defined Networking
SDR	Software Defined Radio
SSID	Service Set Identifier
SSL	Secure Socket Layer
TK	Temporary Key
UAP	Upper Address Part
UE	User Equipment
UI	User Interface
USB	Universal Serial Bus

WEP	Wired Equivalent Privacy
WLAN	Wireless Local Area Network
WPA	Wi-Fi Protected Access
WPS	Wi-Fi Protected Setup
Wi-Fi	Wireless Fidelity
WiGLE	Wireless Geographic Logging Engine

Chapter 1

Introduction

1.1 Motivation

Almost all user equipment with wireless capabilities are created with hardware chips in-bedded with a MAC-address. This MAC-address is unique as each supplier has their own range to choose from. The reason for a MAC-address to be unique is to separate UE in a LAN. UE has been known to broadcast their MAC-address when searching for potential hot spots, access points or pairing devices. Which opens the ability for clandestine identification and location tracking, which is a huge privacy concern. Vendors like Apple and Google care about their users privacy, thus publishing updates regularly to mitigate weaknesses made aware by the security community. In august 2018, one such weakness was made public by the researcher *nightwatchcyber* in his article [5], where he stated *Sensitive Data Exposure via Wi-Fi Broadcasts in Android OS [CVE-2018-9489].* In my specialization project spring 2018, *TTM4502 Telematics Specialization Project Markus Andersen (2018)*, same topic was discussed and the same exposure was discovered.

On the 9th of January 2019, the German police wanted help from the public in finding parcel bomber with MAC-address [6]. The police released a MAC-address (see Figure 1.1) in hope of gathering information related to a Motorola cell phone used by a DHL blackmailer in multiple parcel bomb attacks last year.

Figure 1.1: German Police released photo of wanted DHL parcel bomber with MAC address.

This reflects the idea of active and clandestine identification, and that all devices with hardware chips in-bedded with a MAC-address, creates digital traces in our everyday life.

1.2 Scope and Objectives

In this section the scope will be discussed and objectives listed.

1.2.1 Scope

The thesis will attempt clandestine and active location tracking as well as MiTM attacks against the protocols Wi-Fi and Bluetooth. The thesis will not cover the details of subspace tracking and mesh network tracking as this is thoroughly documented in [7] and [8] respectively.

1.2.2 Objectives

The focus of this thesis is sniffing MAC-addresses from Bluetooth and Wi-Fi. This thesis aim to provide a wide technical background of methods used for sniffing related to privacy issues such as sensitive data exposure. This information is then used to set up and carry out suitable experiments and trials for the purpose of proving such privacy attacks exists. There are four main objectives in this thesis:

Table 1.1: Objectives for this thesis.

1. Build and configure passive Bluetooth and Wi-Fi-sniffing to catch MAC-addresses of UE.
2. Build and configure active Bluetooth and Wi-Fi-sniffing to catch MAC-addresses of UE.
3. Experiment with MiTM-attacks on UE in-bedded with Wi-Fi and Bluetooth.
4. Discuss and analyse the possibilities for improvements and countermeasures to Wi-Fi and Bluetooth in regards to privacy.

In the earlier phases of the thesis, a study of the bluetooth security in hearing aids as well as any problems towards privacy was set as an objective. With multiple attempts of collaboration with different retailers on their hearing aids, the thesis did not receive permission to conduct any testing on hearing aids, thus having to remove this objective. See Subsection 1.7.3 SINTEF: Eavesdropping on Hearing Aids for more information.

1.3 Work Method

This section includes the work method and the implementation of the Cyber Kill Chain (CKC) to highlight the corresponding phases of each method described in later chapters.

1.3.1 Phases

The work method in this thesis is divided in three phases. The first phase included a literature study of the wireless standards, Bluetooth and Wi-Fi, provided by the Wi-Fi Alliance and The Bluetooth Special Interest Group.

The second phase conducted the configuring and experimenting with hardware, software and testing in practical experiments. A clandestine sniffer was built with a Raspberry Pi running Kismet with Wi-Fi adapters, bluetooth adapters and multiple Uberteeth. Active scans and attacks were implemented with Hak5's Pineapple Nano and both Bluetooth adapters and the Uberteeth are used in varying tests.

The third phase of this thesis focuses on mitigation and countermeasures, against proven attacks and possible future attacks that compromise the privacy of any UE.

1.3.2 Cyber Kill Chain

The thesis implements the CKC to categorize the different stages and distinct them apart. The different stages for the thesis, and their implementations, are described in the Table 1.2.

Table 1.2: Cyber Kill Chain.

Reconnaissance	Passively and actively scan for MAC-addresses
Weaponization	Bundle MAC-addresses of targets for location tracking
Delivery	Create fake access point
Exploitation	Gather unencrypted information and redirect target to malicious host
Installation	Install malware on the target asset
Command and Control	Command and control the system remotely
Actions on Objective	Attacker remotely carries out its goals

The CKC is a framework developed by Lockheed Martin used to describe different stages of a cyber attack as it pertains to network security [9]. It is developed mostly to identify and prevent cyber intrusions, but is implemented in this thesis as it suits the attack vectors and help explain the different stages of the attacks.

1.4 Methodology

The thesis will implement the pragmatic methodological approach. This involves using the methods that appears best suited to the research problem, without the need of a philosophical debate about which methods have the best approaches. The pragmatic approach recognises that every method has its own feasibility and limitations, and that the different methods can be complementary. Depending on the method used, the data collected will be analysed in the appropriate manner.

By using the pragmatic approach the thesis should select the best published methods for clandestine and active location tracking, and then establish the feasibility and limitations of those methods by setting up and carrying out suitable experiments and trials for this purpose. As the investigation progresses, the thesis should propose technical and procedural means of avoiding such privacy attacks.

1.5 Contributions

This thesis gives a technical and practical approach of clandestine and active location tracking and eavesdropping on the protocols Wi-Fi and Bluetooth.

The main contribution of this thesis will be a discussion of the feasibility and limitations of existing software and hardware by setting up and carrying out suitable experiments for this purpose. Then propose ways of increasing the privacy of mobile terminals equipped with Wi-Fi and Bluetooth, by providing technical and procedural means of avoiding such privacy attacks.

1.6 Prerequisites and Limitations

For the experiments to be successful there must either exist open source projects that are relevant for the protocols today, or the thesis must create new tools. This thesis has a limited time of only 20 weeks, which gives less time to implement new software if needed. A limitation could be the technicality of existing experiments as these might require technical or theoretical knowledge and understanding. If the literature study is not of high quality with proper source criticism, both the experiments and results would be directly affected. The thesis should choose the best published papers, that are internationally acknowledged or is from known universities, sites or blogs.

1.7 Related Work

In this section the related work towards location tracking, eavesdropping and privacy will be discussed.

1.7.1 Improvements to Androids Wi-Fi

In 2017 Android 8.0 released an update in which the Android devices use a random MAC addresses when probing for new networks while not currently associated to a network [10]. While in Android 9.0, a developer option was implemented to cause the device to use a randomized MAC-address when connecting to a Wi-Fi network, where a different randomized MAC-address is used per Service Set Identifier (SSID). A SSID is commonly known as the name of the wireless router. The security update of November 1st of 2018 changed the bluetooth address, so that there is no correlation between the Wi-Fi and Bluetooth MAC-address.

1.7.2 Improvements to iOS' Wi-Fi

Apple has released similar improvements as Android, as it has implemented MAC-randomization in probe request and removed the SSID [11]. An update on changing the correlation between the Wi-Fi and Bluetooth MAC-address could not be found, making it likely that this is still the case for iOS.

"iOS uses a randomized Media Access Control (MAC) address when conducting Wi-Fi scans while it isn't associated with a Wi-Fi network. These scans could be performed in order to find and connect a preferred Wi-Fi network or to assist Location Services for apps that use geofences, such as location-based reminders or fixing a location in Apple Maps. Note that Wi-Fi scans that happen while trying to connect to a preferred Wi-Fi network aren't randomized."

"iOS also uses a randomized MAC address when conducting enhanced Preferred Network Offload (ePNO) scans when a device isn't associated with a Wi-Fi network or its processor is asleep. ePNO scans are run when a device uses Location Services for apps that use geofences, such as location-based reminders that determine whether the device is near a specific location. Because a device's MAC address now changes when disconnected from a Wi-Fi network, it can't be used to persistently track a device by passive observers of Wi-Fi traffic, even when the device is connected to a cellular network."

"Apple has informed Wi-Fi manufacturers that iOS Wi-Fi scans use a randomized MAC address, and that neither Apple nor manufacturers can predict these randomized MAC addresses. Wi-Fi MAC address randomization support isn't available on iPhone 4s or earlier. On iPhone 6s or later, the hidden property of a known Wi-Fi network is known and updated automatically. If the Service Set Identifier (SSID) of a Wi-Fi network is broadcasted, the iOS device won't send a probe with the SSID included in the request. This prevents the device from broadcasting the network name of non-hidden networks."

1.7.3 SINTEF: Eavesdropping on Hearing Aids

SINTEF did a report on eavesdropping on Hearing Aids on mission from NSM. Their results concluded that it is fairly easy to perform eavesdropping on hearing aids. They conducted excellent research on eavesdropping and suggested multiple attack vectors, where one vector utilized the Bluetooth protocol.

An interview with two of the authors of the report, Are Hellandsvik [12] and Christian Frøystad [13], from the research institution SINTEF was conducted. In the interview, further information on their methods and their conclusions was explained in detail, which gave insight on their outstanding knowledge on the subject.

The interview gave an interesting deep dive into the world of hearing aids, with their implementations and possible security problems. After the interview, the Department of Neuromedicine and Movement Science (INB) at NTNU was contacted to obtain further information on hearing aids. Lars Gunnar Rosvoldaune, who is the study program director at the institute of Audiology, replied with his contacts from the retailers in the industry of hearing aids. After multiple emails and phone calls, the result ended in none of them wanting to collaborate in this thesis, as they all wanted to keep testing and security within their own business and for the most part hidden from the public. This resulted in not receiving permission to conduct any testing on hearing aids, thus having to remove this objective.

1.7.4 How talkative is your mobile device?

In a conducted study from 2015, by Julien Freudiger et al [14], they made an experimental study of Wi-Fi probe requests. They wrote that: "Researchers in previous work have identified privacy hazards associated with Wi-Fi probe requests, such as leaking past access points identifiers and user mobility. Besides several efforts to develop privacy-preserving alternatives, modern mobile devices continue to use Wi-Fi probe requests."

Their objective was to identify how different factors influence the frequency of probe requests and the average number of broadcasted probes. Their conclusions: "On average, some mobile devices send probe requests as often as 55 times per hour, thus revealing their unique MAC address at high frequency. Even if a mobile device is not charging and in sleep mode, it might broadcast about 2000 probes per hour. We also evaluate a commercially deployed MAC address randomization mechanism, and demonstrate a simple method to re-identify anonymized probes."

1.7.5 On Tracking the Physicality of Wi-Fi: A Subspace Approach

In a recent conducted study from 2019, by Mohammed Alloulah et al [7], they made a subspace approach on tracking the physicality of Wi-Fi. "Wi-Fi Channel State Information (CSI) has emerged as a plausible modality for sensing different human activities as a function of modulations in the wireless signal that travels between wireless devices. Until now, most research has taken a statistical approach and/or purpose-built inference pipeline. Although interesting, these approaches struggle to sustain sensing performances beyond experimental conditions. As such, the full potential of CSI as a general-purpose sensing modality is yet to be realized."

They argue that a universal approach with a well-grounded formalization is necessary to characterize the relationship between the wireless channel modulations (spatial and tempral) and human movement. They conclude in their research that their universal channel statistics will uncover opportunities for applying CSI for a variety of human sensing applications in a robust way.

1.8 Discussion

By implementing the pragmatic approach the thesis should be able to bring good results by recognising the feasibility and limitations of the selected methods. Furthermore, the problem description provides a prerequisite that the best published methods for clandestine location tracking should be selected, which requires the methods to be relevant to the versions of the protocols at the time of conducting the experiments. From the problem description we understand that the candidate may construct new methods and propose technical and procedural means of avoiding such privacy attacks. To propose technical and procedural means requires the candidate to have a good understanding of how the protocols work and know of implementations that have been done to avoid such privacy attacks. Furthermore, the thesis should bring a discussion on what technical and procedural means can provide a good overall solution to the privacy attack methods selected.

1.9 Outline

The thesis is divided into a total of 8 chapters, the outline is as follows.

Chapter 1 Introduction
An introduction including motivation, scope and objectives, methodology and related work.

Chapter 2 IEEE 802.11 Wireless
The wireless specification of IEEE 802.11 and relevant theory will be presented.

Chapter 3 Bluetooth
The theory of Bluetooth, Bluetooth Low Energy (BLE) and the possible implementations, in regards to security, will be covered.

Chapter 4 Experiment
Covers the experiments, experimental setup and ways to replicate or reproduce the results.

Chapter 5 Results
Presents the results from experiments and provides a technical explanation to these results.

Chapter 6 Countermeasures
Countermeasures and suggestions of improvement to the protocols Wi-Fi and Bluetooth will be covered.

Chapter 7 Discussion
Provides a discussion on the importance of privacy and the possible impact and attack vectors of the results.

Chapter 8 Conclusion
Presents a conclusion of the work done in this thesis and possible future work.

Chapter 2

IEEE 802.11 Wireless

This chapter includes a general background of the parts of the IEEE 802.11 standard that are needed to understand the content of this thesis.

2.1 Overview

A description of the MAC-address is included as well as the Wired Equivalent Privacy (WEP) and Wi-Fi Protected Access (WPA) protocols. The last part will provide an introduction to probe sniffing, *Wardriving* and access point replicating.

2.2 IEEE 802.11

The IEEE 802.11 is a standard created by the Wi-Fi Alliance® and specifies the protocols for implementing Wireless Local Area Network (WLAN) over various frequencies, including the 2.4 and 5 Gigahertz (GHz) frequency bands [15]. This thesis will focus on the 2.4 GHz frequency band, which includes both Wi-Fi and Bluetooth connections. Last year the release of WPA3 was announced [16], which greatly improves the security compared to WPA2, see Subsection 2.2.4 WPA3-Personal.

2.2.1 Probe Request

The probe requests are usually sent on the all the 13 channels that are most commonly used in Wi-Fi. In Figure 2.1 the different fields in the headers can be seen, where the frame body can vary as there are optional fields that can be added if needed. The header of a probe request gives information about the UE as it contains supported rates, which will be used when sending a probe response and in later stages during the connection request.

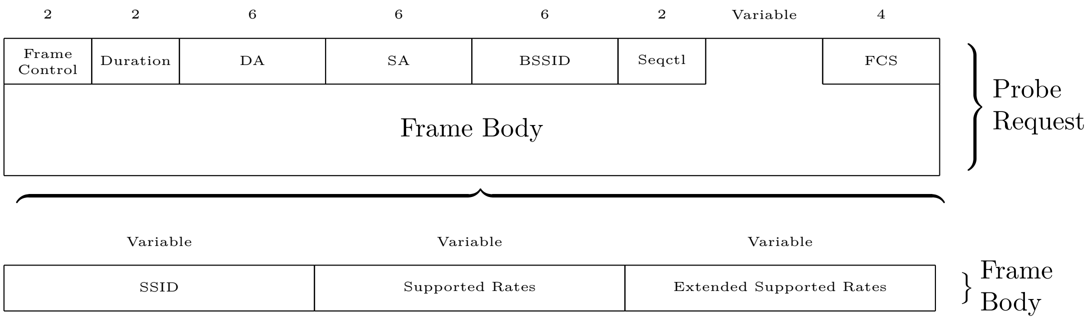


Figure 2.1: Probe request frame with associated sizes.

In Figure 2.2 we have an example broadcast probe request containing information such as supported rates, current channel, source address and even the destination address. In this case the destination is set to broadcast, but this can be changed to the real destination including its MAC-address (Basic Service Set Identifier (BSSID)).

```
Type/Subtype: Probe Request (0x04)
Frame Control: 0x0040 (Normal)
Duration: 0 microseconds
Destination Address: Broadcast (ff:ff:ff:ff:ff:ff)
Source Address: Markus (00:1a:2b:3c:4d:5e)
BSS ID: Broadcast (ff:ff:ff:ff:ff:ff)
Fragment Number: 0
Sequence Number: 1
SSID Parameter Set: Broadcast
Supported Rates: 1.0 2.0 5.5 11.0 6.0 9.0 12.0 18.0
Extended Supported Rates: 24.0 36.0 48.0 54.0
DS Parameter Set: Current Channel: 1
```

Figure 2.2: Probe broadcast request as it would be viewed in Wireshark.

2.2.2 Beacon And Probe Response Frame

The beacon or response frame sent by the AP or routers can be seen in Figure 2.3. The frame contains much more information than just the probe request.

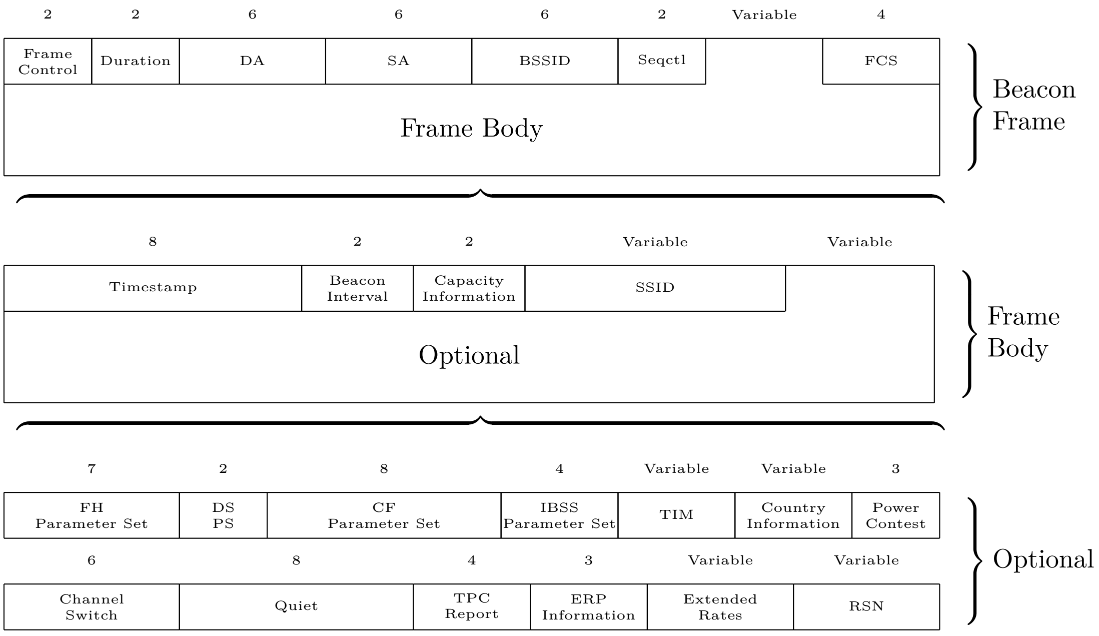


Figure 2.3: Probe response frame with associated sizes.

The probe response gives the UE information about its presence by returning the SSID and extra fields such as the security protocols that it supports. In Figure 2.4 we can read from the field *Vendor Specific* that this router only supports WPA2, which is the most common security setting as of writing this thesis.

```
Type/Subtype: Probe Response (0x05)
Frame Control: 0x0850 (Normal)
Duration: 0
Destination Address: Markus (00:1a:2b:3c:4d:5e)
Source Address: Router (1a:2b:3c:4d:5e:6f)
BSS ID: Router (1a:2b:3c:4d:5e:6f)
Fragment Number: 0
Sequence Number: 1
Timestamp: 0x0000000001234ABC
Beacon Interval: 0.1024
Capacity Information: 0x0512
SSID Parameter Set: "Hide Your Kids Hide Your Wi-Fi"
Supported Rates: 1.0 2.0 5.5 11.0 6.0 9.0 12.0 18.0
Extended Supported Rates: 24.0 36.0 48.0 54.0
DS Parameter Set: Current Channel: 1
ERP Information: no Non-ERP STAs
Vendor Specific: WPA2
```

Figure 2.4: Probe response as it would be viewed in Wireshark.

2.2.3 IPv6

Local IPv6 is often created from the UE's Wi-Fi MAC-address. To convert the MAC-address ***00:1a:2b:3c:4d:5e*** the formula is as follows. First we add *fe80* and then flip the second to last bit of the first byte, which makes ***00*** become *02*. Then we add the next byte *1a*. Between the third and fourth byte we add *ff:fe*, which gives us ***2bff:fe3b***. Then in the end we just add the last two bytes ***4d5e***. The result can be seen in Figure 2.5.

```
Wi-Fi MAC: 00:1a:2b:3c:4d:5e
Local IPv6: fe80::021a:2bff:fe3c:4d5e
```

Figure 2.5: Local IPv6 inherits the MAC-address of the UE.

2.2.4 WPA3-Personal

With WPA3-Personal comes great improvements to the 14 year old WPA2 protocol. From the old password technology known as the Preshared Key (PSK) in WPA2, comes the improvement of Simultaneous Authentication of Equals (SAE) [17]. The technology is resistant to offline dictionary attacks, to guess the PSK of the AP. Thus making all attempts of guessing the PSK to communicate directly with the AP, making it much more time consuming and could easily be discovered and blocked.

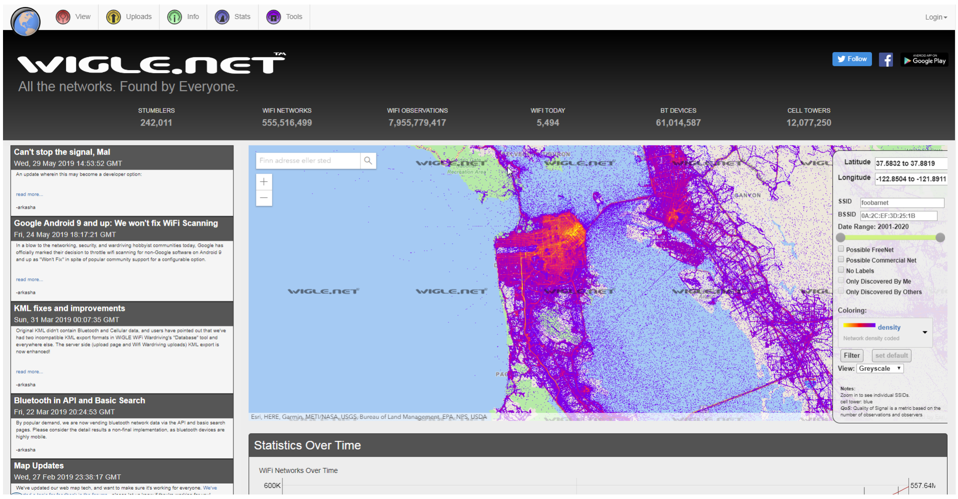
Another great improvement is that the new implementation provides forward secrecy. This means that the data traffic is still protected even if the PSK is compromised after the data has been transmitted.

2.3 Wi-Fi Probe Sniffing

The wireless protocol was created with monitoring mode implemented, which means no special hardware is needed. Suitable software will be covered in Chapter 4 Experiment together with setup, whilst the results are covered in Chapter 5 Results.

Probe requests are usually sent in a interval differentiating from seconds to minutes as set by the UE [14]. This will have an impact of how long an attacker would need to sniff probes in order to decode the MAC-address and to capture the SSIDs sent from the UE. European Union Agency for Network and Information Security (ENISA) wrote back in 2015 about "Abusing Wi-Fi probe requests" and "Profiling based on leaked location information" [18]. Their recommendation was on home and corporate networks not to hide their SSID, as the BSSID will still be broadcasted. They recommended leaving the router settings as defined by the default factory setting, and change the SSID into something unidentifiable [14].

Search engines like Google and Wireless Geographic Logging Engine (WiGLE) have their own database containing millions of access points and their locations. WiGLE is an open source tool containing network locations added by *Wardrivers*. *Wardrivers* are individuals driving through different locations with GPS-equipment logging the locations of wireless networks. These logs are then uploaded to open source libraries like WiGLE that are free to use by the public [19], as can be seen in Figure 2.6.



Can't stop the signal, Mal
Wed, 29 May 2019 14:53:52 GMT
An update wherein this may become a developer option:
read more...
-arkasha

Google Android 9 and up: We won't fix WiFi Scanning
Fri, 24 May 2019 18:17:21 GMT
In a blow to the networking, security, and wardriving hobbyist communities today, Google has officially marked their decision to throttle wifi scanning for non-Google software on Android 9 and up as "Won't Fix" in spite of popular community support for a configurable option.
read more...
-arkasha

KML fixes and improvements
Sun, 31 Mar 2019 00:07:35 GMT
Original KML didn't contain Bluetooth and Cellular data, and users have pointed out that we've had two incompatible KML export formats in WiGLE WiFi Wardriving's "Database" tool and everywhere else. The server side (upload page and Wifi Wardriving uploads) KML export is now enhanced!
read more...
-arkasha

Bluetooth in API and Basic Search
Fri, 22 Mar 2019 20:24:53 GMT
By popular demand, we are now vending bluetooth network data via the API and basic search pages. Please consider the detail results a non-final implementation, as bluetooth devices are highly mobile.
-arkasha

Map Updates
Wed, 27 Feb 2019 23:38:17 GMT
We've updated our web map tech, and want to make sure it's working for everyone. We've

Latitude: 37.5832 to 37.8819
Longitude: -122.8504 to -121.8911
SSID: foobarnet
BSSID: 0A:2C:EF:3D:25:1B
Date Range: 2001-2020
☐ Possible FreeNet
☐ Possible Commercial Net
☐ No Labels
☐ Only Discovered By Me
☐ Only Discovered By Others
Coloring: density — Network density coded
Filter · set default
View: Greyscale

Notes:
Zoom in to see individual SSIDs.
cell tower: blue
QoS: Quality of Signal is a metric based on the number of observations and observers

Esri, HERE, Garmin, METI/NASA, USGS, Bureau of Land Management, EPA, NPS, USDA

Figure 2.6: Screenshot of *wigle.net* (Accessed 2019-05-31).

In April 2019 *Threatpost* wrote about a Chinese app that spilled 2 million hotspot passwords, because of a insecure database [20]. This could potentially result in compromised networks, even home routers and those found in the workplace, if combined with the locations stored by the search engines.

In an article from 2016, *The Hacker News* wrote that "Wi-Fi can be turned into International Mobile Subscriber Identity (IMSI) catcher to track cell phone users everywhere" [21]. This means that wireless routers can capture the IMSI of regular UE and ultimately track and possibly identifying individuals. This information will not be proved by the thesis, as the objectives only specify Wi-Fi and Bluetooth.

2.4 Access Point Replicating

Replicating open and publicly known access points, also know as the *Evil Twin Attack*, could potentially leak private information. This is because the UE will unknowingly connect to a fake access point, which is pretending to be a publicly known AP, the UE will then start synchronizing data. Any unencrypted traffic could be analysed and give more information about the target or target equipment. The AP could potentially redirect traffic and through social and technical engineering make the target download and run malware. This can be done by injecting JavaScript into any website without Secure Socket Layer (SSL) or Hypertext Transfer Protocol Secure (HTTPS) implemented. Even if this is implemented, the fake AP can try to *downgrade* the connection to Hypertext Transfer Protocol (HTTP) if HTTPS is not enforced.

Chapter 3

Bluetooth

Bluetooth is a wireless technology designed for exchanging data over a short distance and is found in most handheld devices today. The technology is most known for connecting UE together to use for hands-free or to stream music to a speaker or headset.

3.1 Bluetooth Classic

Bluetooth classic uses the 2.4 GHz wireless band with 79 designated channels that are 1 MHz wide, which are used in frequency-hopping [22]. With AFH enabled, it usually performs about 1600 hops per second. This is a security mechanism to complicate the process of sniffing on active connections.

3.1.1 Bluetooth Connection Handshake

From Figure 3.1 we can observe the different phases of a connection handshake. The handshake first creates a Temporary Key (TK), a six digit Passkey from the value *000000* to *999999*, before calculating the Long Term Key (LTK).

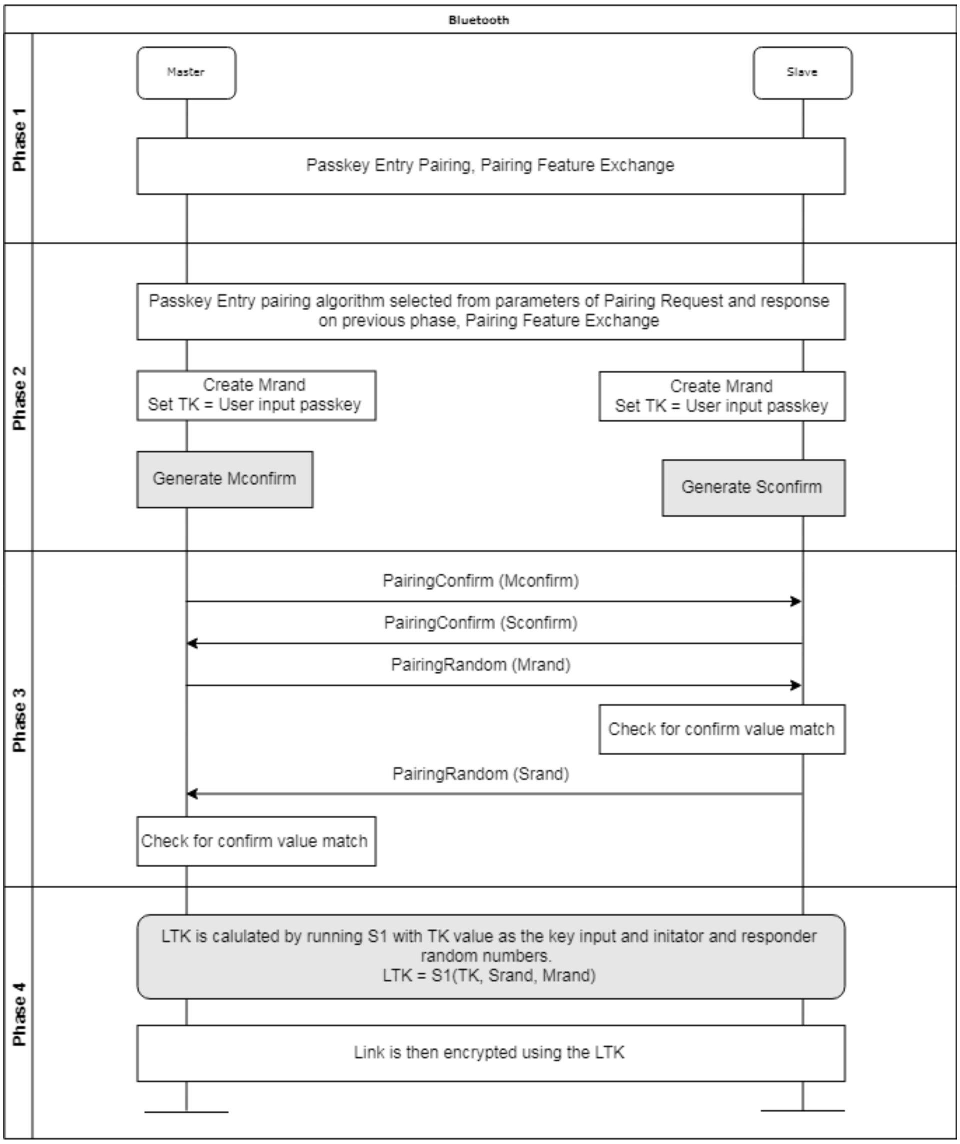

Bluetooth
Phase 1
Phase 2
Phase 3
Phase 4
Master
Slave
Passkey Entry Pairing, Pairing Feature Exchange
Passkey Entry pairing algorithm selected from parameters of Pairing Request and response on previous phase, Pairing Feature Exchange
Create Mrand
Set TK = User input passkey
Create Mrand
Set TK = User input passkey
Generate Mconfirm
Generate Sconfirm
PairingConfirm (Mconfirm)
PairingConfirm (Sconfirm)
PairingRandom (Mrand)
Check for confirm value match
PairingRandom (Srand)
Check for confirm value match
LTK is calulated by running S1 with TK value as the key input and initator and responder random numbers.
LTK = S1(TK, Srand, Mrand)
Link is then encrypted using the LTK

Figure 3.1: Bluetooth connection handshake.

3.2 Bluetooth Low Energy

In BLE the channels are 2 MHz wide, from the 1 MHz in regular Bluetooth, thus only having space for 40 designated channels. BLE is designed to make sniffing difficult as it uses 3 separate channels for advertising, uses the Frequency Hopping Spread Spectrum (FHSS) and both devices can renegotiate some parameters at any given time.

3.2.1 Security Modes

There are four security levels appropriately numbered 1 through 4, with 4 being the most secure, see Table 3.2. There are two security modes: LE Security Mode 1 and LE Security Mode 2, see Table 3.1. To further complicate things, there are two additional security modes named Mixed Security Mode and Secure Connection Only Mode [23]. The security levels 2 and 3 has Advanced Encryption Standard - Cipher-based Message Authentication Code (AES-CMAC) encryption implemented, while security level 4 has Elliptic-Curve Diffie-Hellman (ECDH) encryption. The difference is that ECDH requires smaller keys to achieve the same level of security as the AES-CMAC. This makes the protocol more effective as the goal of BLE protocol is to use as little energy as possible, while having the option of good security and encryption.

Table 3.1: Bluetooth Low Energy Security Levels.

Security Level 1
Supports communication without security at all
Applies to any unpaired Bluetooth communication

Security Level 2
Unpaired communication
AES-CMAC encryption

Security Level 3
Requires paired communication
AES-CMAC encryption

Security Level 4
Requires paired communication
ECDH encryption

Table 3.2: Bluetooth Low Energy Security Modes.

Security Mode 1 ■
Supports all levels where data is unsigned

Security Mode 2 ☑
Supports all levels where data is signed

Mixed Security Mode ■ ☑
Supports all levels with both signed and unsigned data

🔒 **Secure Connection Only Mode** ■ ☑
Supports Security Level 4 with both signed and unsigned data

3.3 Bluetooth MAC-address

The MAC-address consists of 48 bits, example **11:22:33:44:55**, and the address denotes several pieces of information [24].

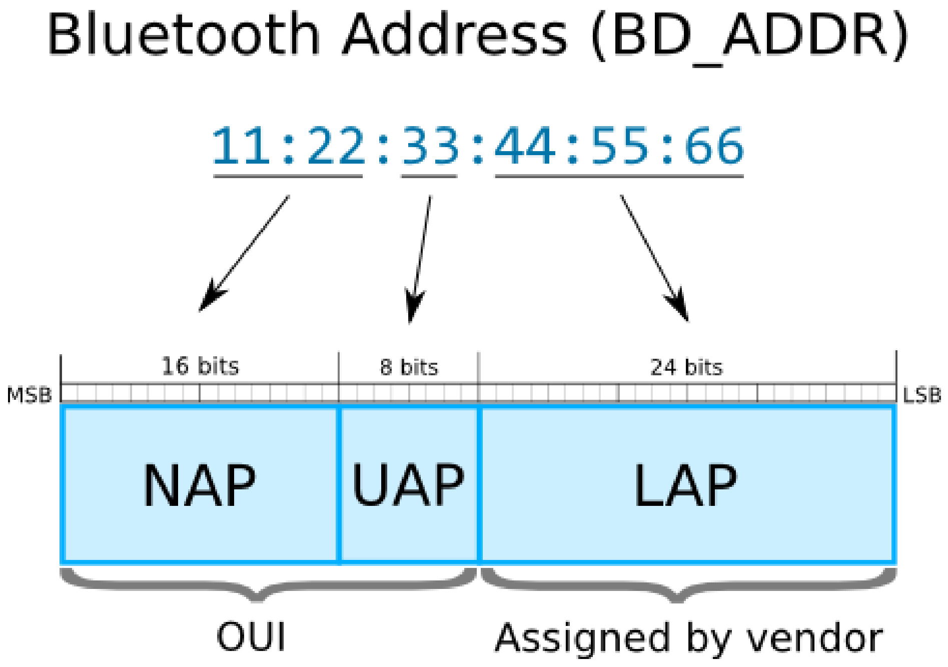


Figure 3.2: The different parts of the MAC-address [3] (Accessed 2019-05-19).

- Non-significant Address Part (NAP), is the first 16 bits of the address, **11**:**22**:33:44:55, which is used in the FHSS frames.

- UAP, is the next 8 bits of the address, 11:22:**33**:44:55, which is used for seeding in various Bluetooth specification algorithms.
- Organizationally Unique Identifier (OUI), is the first 24 bits of the address, **11**:**22**:**33**:44:55, which is a combination of the NAP and the UAP. The OUI reveals the manufacturer of the device and can be found using a OUI-lookup table.
- LAP, is the last 24 bits of the address, 11:22:33:**44**:**55**, which is allocated by the vendor of the device. The LAP identifies the Bluetooth device and is transmitted with each packet as part of the packet header.

The MAC-address can be uniquely identifiable, with a fairly high certainty, having only the last 32 bits of the 48 bit address.

Chapter 4

Experiment

In this chapter the experimental setup is discussed together with explanations of problems and solutions, and what became the method of choice in the end.

4.1 Experimental Setup

In this subsection the experimental setup of every hardware and software that was considered for testing is explained with a step-by-step guide. All hardware used in the experiments can be seen in Figure 4.1.

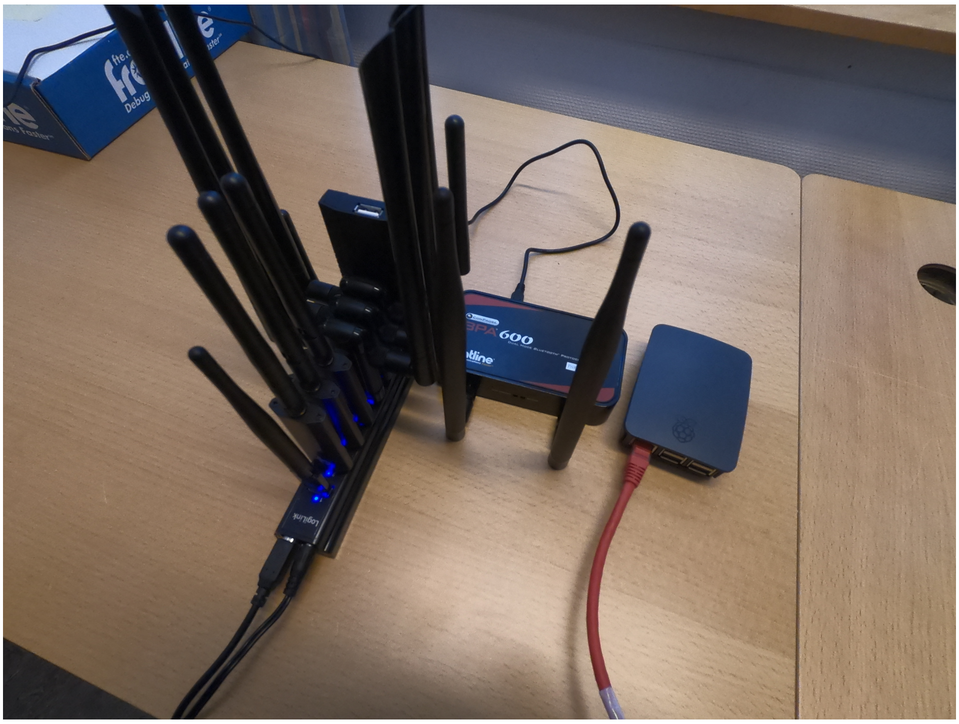


Figure 4.1: Hardware used in the experiment.

4.1.1 Raspberry Pi

The Raspberry Pi is a small computer which can be used for multiple things. The best is to download software directly from their website [25]. The Raspberry Pi can be seen in Figure 4.2. For this thesis, the Raspberry Pi suits the objective of clandestine location tracking for its size and portability.

Figure 4.2: Raspberry Pi

4.1.2 Alfa USB Wireless Adapter Setup

If your computer does not support the adapters out of the box, you would have to either install the drivers from Alfa (`https://alfa.com.tw`) or update your wireless drivers. The following shows how you install the Alfa Adapter (AWUS036AC) drivers for linux.

```
$ sudo apt install build-essential linux-headers-‘uname -r‘
$ wget https://www.alfa.com.tw/files/%5B1%5D%20WiFi%20USB%20adapter/
    ↪ AWUS036AC/Linux/AWUS036AC_036EAC_ACH_linux_v4.3.2_11100.20140411.tar
$ tar -xvf AWUS036AC_036EAC_ACH_linux_v4.3.2_11100.20140411.tar
$ rm AWUS036AC_036EAC_ACH_linux_v4.3.2_11100.20140411.tar
$ cd AWUS036AC_036EAC_ACH_linux_v4.3.2_11100.20140411
$ make
$ sudo make install
$ reboot
```

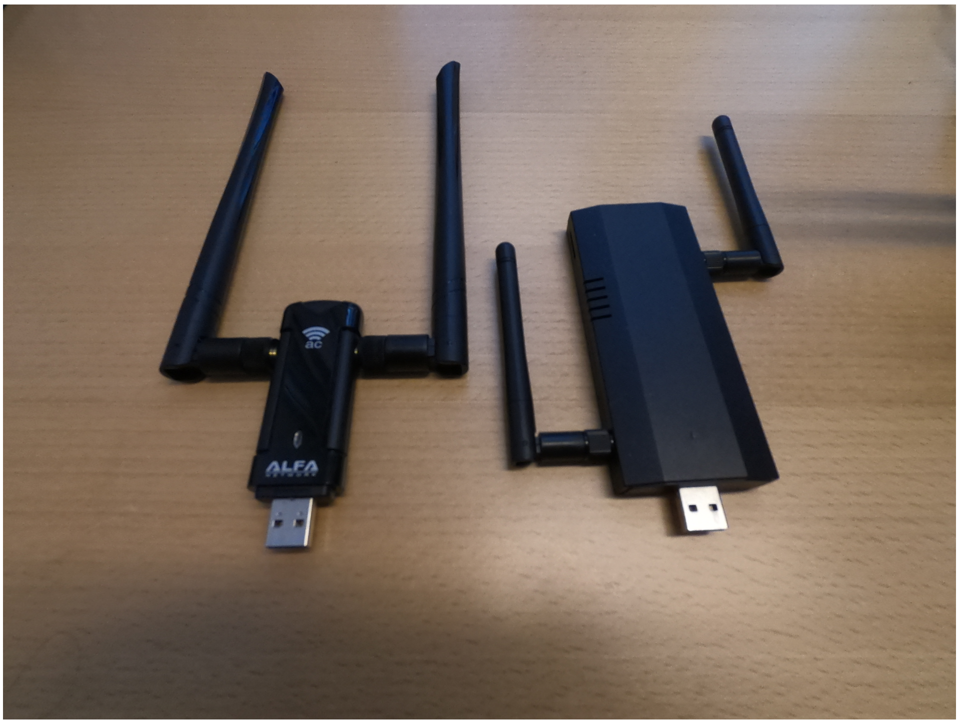


Figure 4.3: Alfa Wireless Adapter (left) and Hak5 WiFi Pineapple Nano (right).

4.1.3 Hak5 WiFi Pineapple Setup

The Wi-Fi Pineapple Nano can be seen to the right in Figure 4.3. Information about the device can be found on the Hak5 website (`https://shop.hak5.org/products/wifi-pineapple`). The Pineapple is a device that support Plug-and-Play (PnP), which makes it easy to use and to setup. After the device is connected into the Universal Serial Bus (USB)-port visit the following website using your favorite browser.

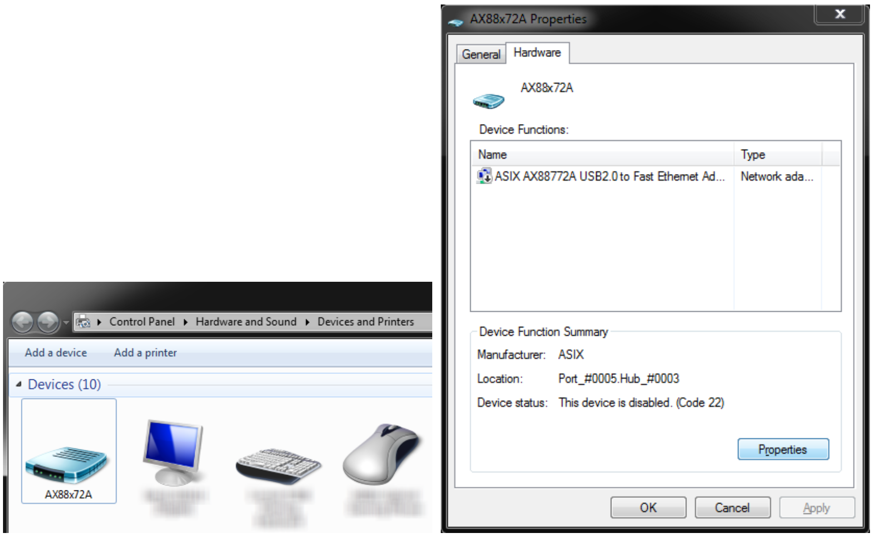
```
http://172.16.42.1:1471
```

The Graphical User Interface (GUI) is easy to use and understand as can be seen in Figure 5.7. From Table 4.1, we see that the Pineapple has the following features.

Table 4.1: Features of Hak5's Pineapple.

- Leading Rogue Access Point
  Patented PineAP Suite thoroughly mimics preferred networks, enabling man-in-the-middle attacks
- WPA and WPA Enterprise Attacks
  Capture WPA handshakes and imitate enterprise access points, capturing enterprise credentials
- Precision Targeting Filters
  Stay within the scope of engagement and limit collateral damage with MAC and SSID filtering
- Simple Web Interface
  Fast and intuitive with an emphasis on workflow and actionable intelligence – just click to attack
- Cross-Platform
  No software to install. Works in any modern web browser on Windows, Mac, Linux, Android, iOS
- Advanced Reconnaissance
  Visualize the WiFi landscape and the relationships between access points and devices
- Actionable Intelligence
  Identify vulnerable devices, gather intelligence on the target and direct attacks
- Passive Surveillance
  Monitor and collect data from all devices in the vicinity. Save and recall reports at any time
- Active Frame Injection Attacks
  Perform targeted, active WiFi attacks with manipulated management frames including deauth
- Cloud C Enabled
  Deploy with confidence. Remotely command and control the airwaves with Hak5 Cloud C

4.1.4 Hak5 WiFi Pineapple Windows 7 Setup


Control Panel ▸ Hardware and Sound ▸ Devices and Printers

Add a device   Add a printer

Devices (10)

AX88x72A

AX88x72A Properties

General | Hardware

AX88x72A

Device Functions:

Name	Type
ASIX AX88772A USB2.0 to Fast Ethernet Ad...	Network ada...

Device Function Summary

Manufacturer: ASIX

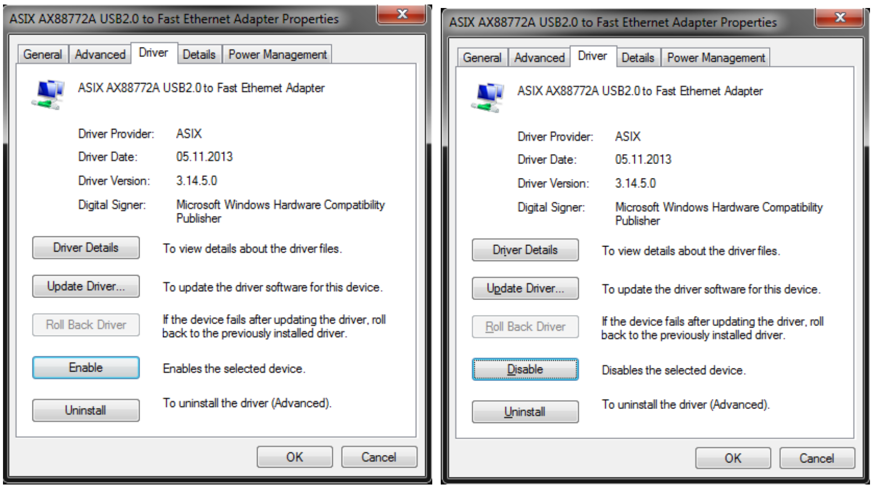
Location: Port_#0005.Hub_#0003

Device status: This device is disabled. (Code 22)

Properties

OK   Cancel   Apply

Figure 4.4: Devices and printers, Properties.


ASIX AX88772A USB2.0 to Fast Ethernet Adapter Properties

General | Advanced | Driver | Details | Power Management

ASIX AX88772A USB2.0 to Fast Ethernet Adapter

Driver Provider:	ASIX
Driver Date:	05.11.2013
Driver Version:	3.14.5.0
Digital Signer:	Microsoft Windows Hardware Compatibility Publisher

Driver Details	To view details about the driver files.
Update Driver...	To update the driver software for this device.
Roll Back Driver	If the device fails after updating the driver, roll back to the previously installed driver.
Enable	Enables the selected device.
Uninstall	To uninstall the driver (Advanced).

OK   Cancel

ASIX AX88772A USB2.0 to Fast Ethernet Adapter Properties

General | Advanced | Driver | Details | Power Management

ASIX AX88772A USB2.0 to Fast Ethernet Adapter

Driver Provider:	ASIX
Driver Date:	05.11.2013
Driver Version:	3.14.5.0
Digital Signer:	Microsoft Windows Hardware Compatibility Publisher

Driver Details	To view details about the driver files.
Update Driver...	To update the driver software for this device.
Roll Back Driver	If the device fails after updating the driver, roll back to the previously installed driver.
Disable	Disables the selected device.
Uninstall	To uninstall the driver (Advanced).

OK   Cancel

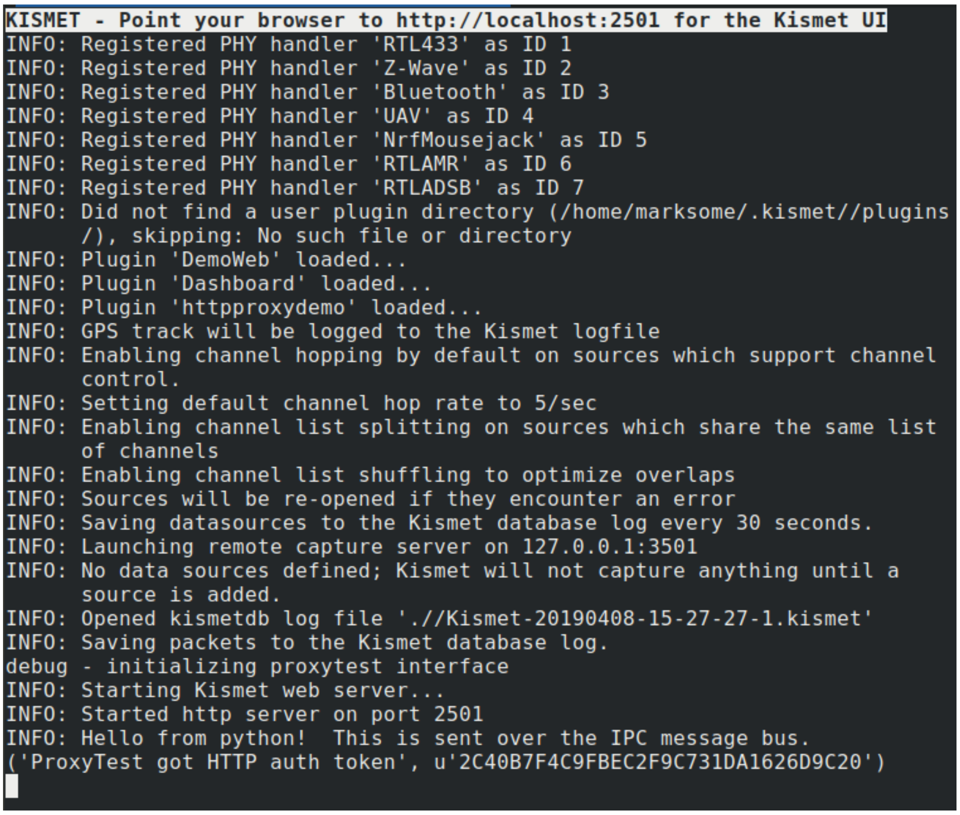
Figure 4.5: Enable driver.

4.1.5 Kismet Setup

Explanation to setup and information about the kismet can be found here [26]. The kismet is a open source software to monitor and collect MAC-addresses of nearby devices using the chosen hardware connected to the computer. Following is the commands used to setup the kismet for first time use.

```
$ wget https://www.kismetwireless.net/code/kismet-2019-04-R1.tar.xz
$ tar -xvf kismet-2019-04-R1.tar.xz
$ rm kismet-2019-04-R1.tar.xz
$ cd kismet-2019-04-R1
$ ./configure
$ make
$ sudo make suidinstall
$ sudo usermod -a -G kismet $USER
$ kismet
```

After running the kismet program you will receive info about pointing you browser to *http://localhost:2501* for the Kismet User Interface (UI).



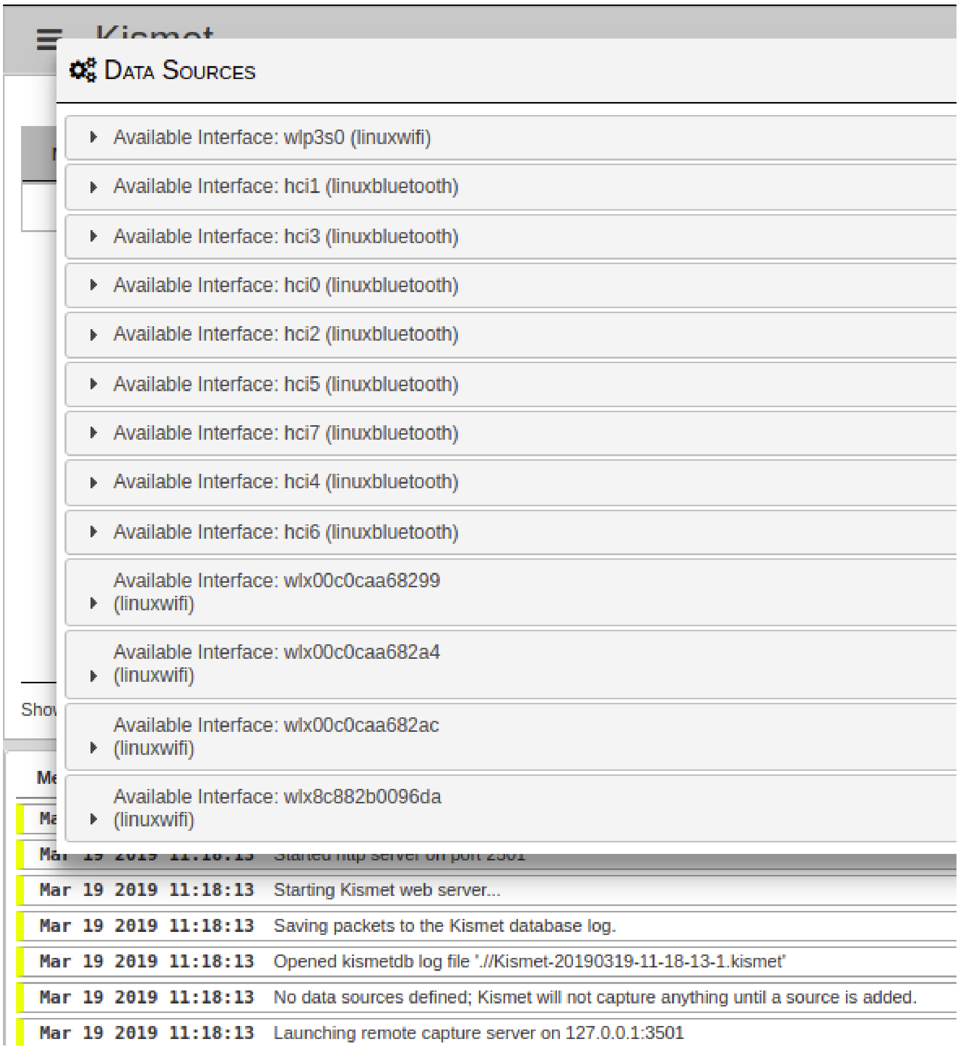
```
KISMET - Point your browser to http://localhost:2501 for the Kismet UI
INFO: Registered PHY handler 'RTL433' as ID 1
INFO: Registered PHY handler 'Z-Wave' as ID 2
INFO: Registered PHY handler 'Bluetooth' as ID 3
INFO: Registered PHY handler 'UAV' as ID 4
INFO: Registered PHY handler 'NrfMousejack' as ID 5
INFO: Registered PHY handler 'RTLAMR' as ID 6
INFO: Registered PHY handler 'RTLADSB' as ID 7
INFO: Did not find a user plugin directory (/home/marksome/.kismet//plugins
      /), skipping: No such file or directory
INFO: Plugin 'DemoWeb' loaded...
INFO: Plugin 'Dashboard' loaded...
INFO: Plugin 'httpproxydemo' loaded...
INFO: GPS track will be logged to the Kismet logfile
INFO: Enabling channel hopping by default on sources which support channel
      control.
INFO: Setting default channel hop rate to 5/sec
INFO: Enabling channel list splitting on sources which share the same list
      of channels
INFO: Enabling channel list shuffling to optimize overlaps
INFO: Sources will be re-opened if they encounter an error
INFO: Saving datasources to the Kismet database log every 30 seconds.
INFO: Launching remote capture server on 127.0.0.1:3501
INFO: No data sources defined; Kismet will not capture anything until a
      source is added.
INFO: Opened kismetdb log file './/Kismet-20190408-15-27-27-1.kismet'
INFO: Saving packets to the Kismet database log.
debug - initializing proxytest interface
INFO: Starting Kismet web server...
INFO: Started http server on port 2501
INFO: Hello from python!  This is sent over the IPC message bus.
('ProxyTest got HTTP auth token', u'2C40B7F4C9FBEC2F9C731DA1626D9C20')
```

Figure 4.6: Kismet running from terminal.

Using your favorite web browser you can visit the following website.

```
http://127.0.0.1:2501/
```

Clicking on *Data Sources* in the menu will let you add sources you want to show MAC-addresses found by the data source.



```
Available Interface: wlp3s0 (linuxwifi)
Available Interface: hci1 (linuxbluetooth)
Available Interface: hci3 (linuxbluetooth)
Available Interface: hci0 (linuxbluetooth)
Available Interface: hci2 (linuxbluetooth)
Available Interface: hci5 (linuxbluetooth)
Available Interface: hci7 (linuxbluetooth)
Available Interface: hci4 (linuxbluetooth)
Available Interface: hci6 (linuxbluetooth)
Available Interface: wlx00c0caa68299 (linuxwifi)
Available Interface: wlx00c0caa682a4 (linuxwifi)
Available Interface: wlx00c0caa682ac (linuxwifi)
Available Interface: wlx8c882b0096da (linuxwifi)
```

```
Mar 19 2019 11:18:13  Starting Kismet web server...
Mar 19 2019 11:18:13  Saving packets to the Kismet database log.
Mar 19 2019 11:18:13  Opened kismetdb log file './/Kismet-20190319-11-18-13-1.kismet'
Mar 19 2019 11:18:13  No data sources defined; Kismet will not capture anything until a source is added.
Mar 19 2019 11:18:13  Launching remote capture server on 127.0.0.1:3501
```

Figure 4.7: Kismet Data Sources shown in web browser.

After adding your data sources you will instantly notice active devices displayed, if they can be found nearby of course.

4.1.6 Ubertooth Setup

The Ubertooth, which can be seen to the right in Figure 4.8, is a bluetooth USB adapter that is specifically designed for monitoring bluetooth signals [27]. The ubertooth is a passive adapter, which compared to Wi-Fi, is the same setting the wireless adapter to monitor mode. Before we install *ubertooth-rx* package, we need to first install the prerequisites by pasting the following command into the terminal.

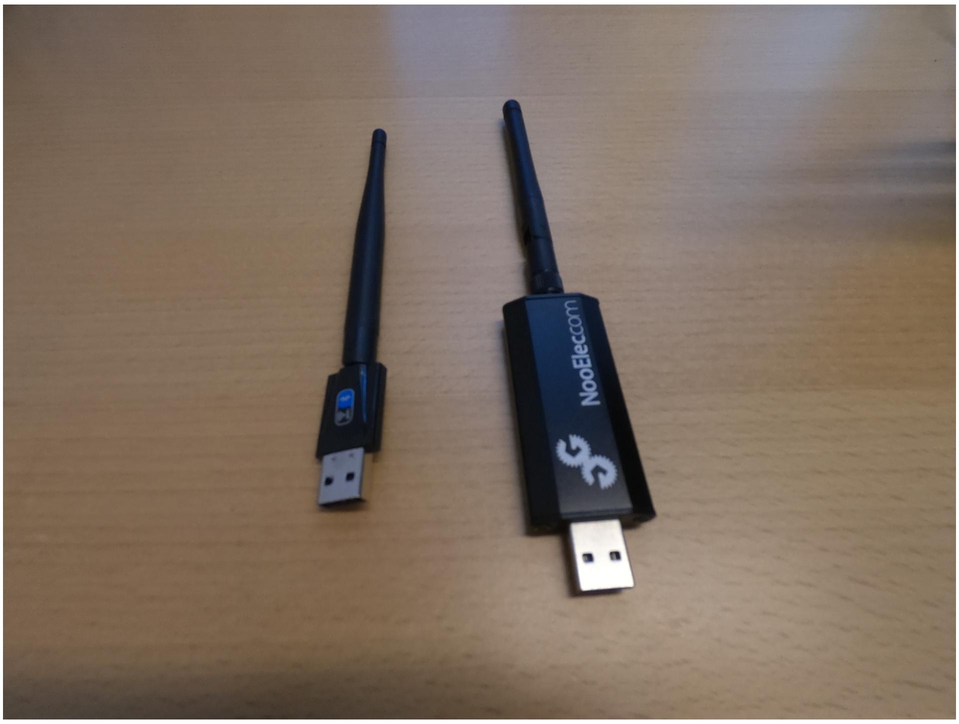
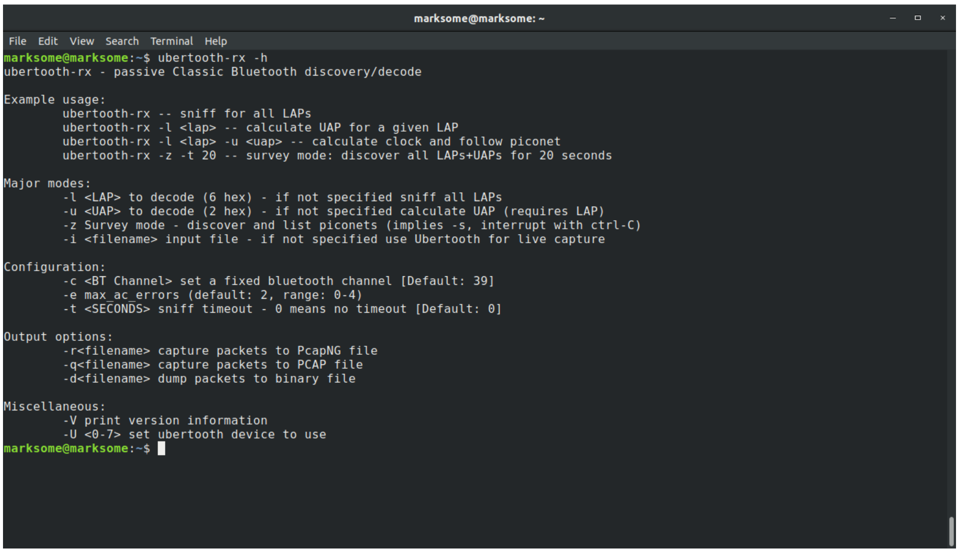


Figure 4.8: Bluetooth wireless adapter (left) and Ubertooth (right).



```
marksome@marksome:~$ ubertooth-rx -h
ubertooth-rx - passive Classic Bluetooth discovery/decode

Example usage:
        ubertooth-rx -- sniff for all LAPs
        ubertooth-rx -l <lap> -- calculate UAP for a given LAP
        ubertooth-rx -l <lap> -u <uap> -- calculate clock and follow piconet
        ubertooth-rx -z -t 20 -- survey mode: discover all LAPs+UAPs for 20 seconds

Major modes:
        -l <LAP> to decode (6 hex) - if not specified sniff all LAPs
        -u <UAP> to decode (2 hex) - if not specified calculate UAP (requires LAP)
        -z Survey mode - discover and list piconets (implies -s, interrupt with ctrl-C)
        -i <filename> input file - if not specified use Ubertooth for live capture

Configuration:
        -c <BT Channel> set a fixed bluetooth channel [Default: 39]
        -e max_ac_errors (default: 2, range: 0-4)
        -t <SECONDS> sniff timeout - 0 means no timeout [Default: 0]

Output options:
        -r<filename> capture packets to PcapNG file
        -q<filename> capture packets to PCAP file
        -d<filename> dump packets to binary file

Miscellaneous:
        -V print version information
        -U <0-7> set ubertooth device to use
marksome@marksome:~$
```

Figure 4.9: Ubertooth-rx options (*ubertooth-rx -h*).

```
sudo apt-get install cmake libusb-1.0-0-dev make gcc g++ libbluetooth-dev
    ↪ pkg-config libpcap-dev python-numpy python-pyside python-qt4
```

After the prerequisites are installed we ready to install the ubertooth software. To install *ubertooth-rx* you could download the software from github [28] or run the following commands.

```
wget https://github.com/greatscottgadgets/ubertooth/releases/download
    ↪ /2018-12-R1/ubertooth-2018-12-R1.tar.xz
tar xf ubertooth-2018-12-R1.tar.xz
cd ubertooth-2018-12-R1/host
mkdir build
cd build
cmake ..
make
sudo make install
```

If you are installing for the first time, or you receive errors about finding the library, you should run the following command before executing the make and make install again.

```
sudo ldconfig
```

Now the software should be up and running and good to go. By running the command *ubertooth-rx -h*, the different options and commands are listed for *ubertooth-rx*, as seen in Figure 4.9.

4.1.7 Bettercap Setup

Bettercap is a powerful, easily extensible and portable framework written in Go which aims to offer to security researchers, red teamers and reverse engineers an easy to use, all-in-one solution with all the features they might possibly need for performing reconnaissance and attacking WiFi networks, Bluetooth Low Energy devices, wireless HID devices and Ethernet networks [29].

The following list includes modes of operation including features for the Bettercap software [29].

Table 4.2: Features of the Bettercap software.

- WiFi networks scanning, deauthentication attack, clientless PMKID association attack and automatic WPA/WPA2 client handshakes capture.
- Bluetooth Low Energy devices scanning, characteristics enumeration, reading and writing.
- 2.4Ghz wireless devices scanning and MouseJacking attacks with over-the-air HID frames injection (with DuckyScript support).
- Passive and active IP network hosts probing and recon.
- ARP, DNS and DHCPv6 spoofers for MITM attacks on IP based networks.
- Proxies at packet level, TCP level and HTTP/HTTPS application level fully scriptable with easy to implement javascript plugins.
- A powerful network sniffer for credentials harvesting which can also be used as a network protocol fuzzer.
- A very fast port scanner.
- A powerful REST API with support for asynchronous events notification on websocket to orchestrate your attacks easily.
- A very convenient web UI.
- Modules [30].

The following is a step by step guide, including problems during setup.

```
$ sudo wget https://dl.google.com/go/go1.12.2.linux-amd64.tar.gz
$ sudo tar -C /usr/local -xzf go1.12.2.linux-amd64.tar.gz
$ rm go1.12.2.linux-amd64.tar.gz
$ go get github.com/golang/dep
$ cd $HOME/go/src/github.com/golang/dep
$ go install ./...
$ sudo apt install go-dep
$ sudo apt install libnetfilter-queue-dev
$ export GOPATH=~/go
$ export PATH=$PATH:/usr/local/go/bin:$GOPATH/bin
$ cd ~/go/src
$ go get -u github.com/bettercap/bettercap
$ cd ~/go/src/github.com/bettercap/bettercap
$ sudo make
```

This resulted in an error which is summarized in issue 469 [31].

```
gofmt -s -w core firewall log modules network packets session tls
dep: WARNING: Unknown field in manifest: prune
# github.com/bettercap/bettercap/vendor/github.com/mdlayher/raw
vendor/github.com/mdlayher/raw/raw_linux.go:88:14: f.SyscallConn undefined
    ↪ (type *os.File has no field or method SyscallConn)
Makefile:13: recipe for target 'build' failed
make: *** [build] Error 2
```

The solution chosen was to reset the HEAD to the resolving fix.

```
$ sudo git reset --hard e98ac9938f142759e6f24c28e15284ee21a93c88
```

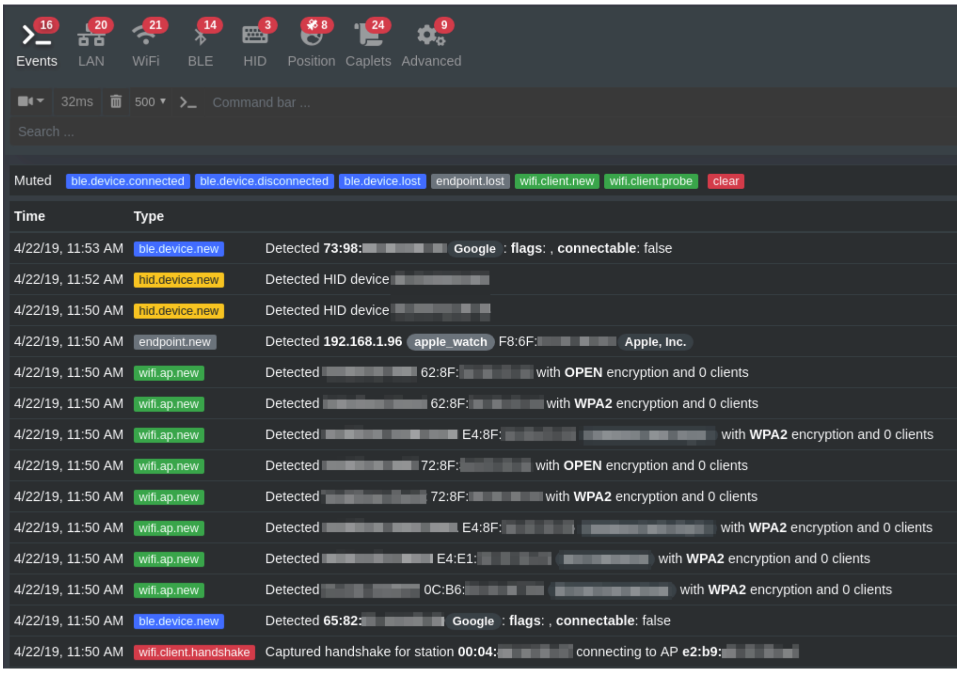
And now you should be able to make and install without problems.

```
$ sudo make
$ sudo make install
```

In Figure 4.10 the current Events are showing with options of choosing tabs containing only LAN, Wi-Fi, BLE and more.

4.1.8 Gattack Setup

For multiple attacks on Bluetooth devices and for bluetooth connections. Gattack can be used for the following modes of operation and usage [32], see Table 4.3.



```
Muted  ble.device.connected  ble.device.disconnected  ble.device.lost  endpoint.lost  wifi.client.new  wifi.client.probe  clear

Time                 Type
4/22/19, 11:53 AM    ble.device.new           Detected 73:98:[redacted] Google : flags: , connectable: false
4/22/19, 11:52 AM    hid.device.new           Detected HID device [redacted]
4/22/19, 11:50 AM    hid.device.new           Detected HID device [redacted]
4/22/19, 11:50 AM    endpoint.new             Detected 192.168.1.96 apple_watch F8:6F:[redacted] Apple, Inc.
4/22/19, 11:50 AM    wifi.ap.new              Detected [redacted] 62:8F:[redacted] with OPEN encryption and 0 clients
4/22/19, 11:50 AM    wifi.ap.new              Detected [redacted] 62:8F:[redacted] with WPA2 encryption and 0 clients
4/22/19, 11:50 AM    wifi.ap.new              Detected [redacted] E4:8F:[redacted] [redacted] with WPA2 encryption and 0 clients
4/22/19, 11:50 AM    wifi.ap.new              Detected [redacted] 72:8F:[redacted] with OPEN encryption and 0 clients
4/22/19, 11:50 AM    wifi.ap.new              Detected [redacted] 72:8F:[redacted] with WPA2 encryption and 0 clients
4/22/19, 11:50 AM    wifi.ap.new              Detected [redacted] E4:8F:[redacted] [redacted] with WPA2 encryption and 0 clients
4/22/19, 11:50 AM    wifi.ap.new              Detected [redacted] E4:E1:[redacted] [redacted] with WPA2 encryption and 0 clients
4/22/19, 11:50 AM    wifi.ap.new              Detected [redacted] 0C:B6:[redacted] [redacted] with WPA2 encryption and 0 clients
4/22/19, 11:50 AM    ble.device.new           Detected 65:82:[redacted] Google : flags: , connectable: false
4/22/19, 11:50 AM    wifi.client.handshake    Captured handshake for station 00:04:[redacted] connecting to AP e2:b9:[redacted]
```

Figure 4.10: Bettercap running with current Events [4].

Table 4.3: Gattack modes of operation.

- disrupting functionality (Denial of Service)
- spoofing (false indications, disabling alarms)
- data interception of (e.g. personal information, authentication etc)
- taking control over the device (e.g. opening smart lock, turning smart home)

To install Gattack we run the following command to install it with Node Package Manager (NPM).

```
npm install gattacker
```

To start a central device that connects to the targeted peripheral and acts as websocket server we run the following command.

```
sudo node ws-slave
```

To scan for advertising devices you need to run this command.

```
node scan
```

For many applications it is necessary to clone MAC address of original device. A helper tool *bdaddr* from *Bluez* is provided in *helpers/bdaddr*, with the wrapper script *mac_adv*.

```
cd helpers/bdaddr
make
./mac_adv -a <advertisement_json_file> [ -s <services_json_file> ]
```

4.1.9 Crackle Setup

The software could be implemented to crack and decrypt the BLE encryption. Crackle has the following modes of operation. Hence, Crackle is a tool for cracking the TK and the LTK from a bluetooth connection [33], see Table 4.4.

Table 4.4: Crackle modes of operation.

- Crack TK
- Crack LTK

The following usage of crackle for cracking the LTK and future communications is a copy from darknet [34].

```
# crack TK mode
$ crackle -i <file.pcap> -o <decrypted.pcap>
TK found: 412741
LTK found: 26db138f0cc63a12dd596228577c4730
Done, processed 306 total packets, decrypted 17

# decrypting future communications with the above LTK
$ crackle -i <file.pcap> -o <decrypted.pcap> -l 26
    ↪ db138f0cc63a12dd596228577c4730
Done, processed 373 total packets, decrypted 15
```

4.2 Other Available Software and Hardware

This section will cover other available software and hardware that was considered for experiments in this thesis, but

4.2.1 Software Defined Radio

Software Defined Networking (SDN)-modules in existing toolkits are only capable of sniffing bluetooth advertisements, this is because specially designed hardware is needed for sniffing encrypted bluetooth signals. Thus, SDN-mdoules are unable to follow any existing/new connections.

4.2.2 Bluefruit LE Sniffer

Bluefruit is created by Adafruit and has proprietary firmware from Nordic Semiconductor [35]. Bluefruit has the capability to sniff new bluetooth connections, and is a cheaper option than Ubertooth One 4.1.6 Ubertooth Setup.

4.2.3 nRF Sniffer

nRF Sniffer is a useful tool for debugging and learning about BLE applications [36]. The *nRF Sniffer* allows near real-time display of Bluetooth packets. To use the *nRF Sniffer* you need either the *nRF52 DK*, *nRF51 DK* or the *nRF51 Dongle*, hence making this software a more expensive option and not chosen for the technical approach.

4.2.4 GNU Radio

GNU's Not Unix (GNU) Radio is a open source software development toolkit that provides signal processing block to implement software radios [37]. This is a more expensive option as it requires a 2.4GHz compatible Software Defined Radio (SDR) device. Even though GNU Radio is much appreciated toolkit and used in research, industry, academia, government, and hobbyist environments, it is not the right tool if you want to sniff or hack bluetooth connections.

4.2.5 Micro:Bit, Btlejuice and Btlejack

For the experiments conducted in this thesis, the hardware *Micro:Bit* was considered together with the software *btlejuice* [38] and *btlejack* [39]. Testing with these software was conducted in the paper *POCORGTFO17* posted by the site *alchemistowl* in 2017 [40]. These soft- and hardwares were not tested during the experiments, but would be recommended for future testing.

4.3 Method of Choice Obtaining MAC-addresses and MiTM-attacks

During the selection of methods, the thesis had every intention of affordable, Public-of-the-Shelf (POTS), and portable solutions. The idea was ability to carry the equipment in a small bag, to be able to carry out location tracking of devices at any site suited or desirable. All methods should give the option of using a simple Raspberry Pi or mobile device.

4.3.1 Passive Wi-Fi Sniffing

For passive sniffing on Wi-Fi, the method of choice ended up on using three Alfa USB Wireless Adapters (see 4.1.2 Alfa USB Wireless Adapter Setup) together with Kismet (see 4.1.5 Kismet Setup) to sniff on the 13 different channels that are commonly used in Wi-Fi. The result is shown in figure 5.2.

The reason for these choices came down to the ease of access to the hardware, POTS, combined with a simple setup and functionality. Although the Alfa adapters proved the need of drivers, it was easy to find and install. Kismet, with its latest release at the time (kismet-2019-04-R1), came with a graphical interface that replaced the old terminal operating window.

4.3.2 Passive Bluetooth Sniffing

In passive bluetooth sniffing, the possibilities where few as bluetooth requires specifically designed hardware, as to Wi-Fi which has monitor-mode in-bedded in its design. Nordic Semiconductor has made multiple hardware solutions, but these proved to be costly and not part of the POTS requirement. Thus, resulting in the choice of Ubertooth as this option is cheaper and both easy to setup and use, see Section 4.1.6 Ubertooth Setup.

For software the options was sorted down to only two, the 4.1.7 Bettercap Setup and the 4.1.5 Kismet Setup. Both software was used in the experiments, but the best results where reached with Kismet.

4.3.3 Active Wi-Fi Sniffing and MiTM-attack

For active sniffing on Wi-Fi, the choice came down to the Hak5 Pineapple Nano (see 4.1.3) as the device is plug-and-play. This meaning that you can simply plug the device into any linux-based system and visit *http://localhost:2501* in your browser. Pineapple supports Windows, MacOS, Linux and even android. This piece of hardware comes with multiple options and modes of operation [41], seen in Table 4.5.

Table 4.5: Features of the Hak5's Pinapple Nano.

1. Perform advanced man-in-the-middle attacks by thoroughly mimicking preferred networks
2. Gather intelligence, including what other networks the targets have connected
3. Completely visualizes the WiFi landscape with continuous, live passive monitoring
4. Capture WPA and WPA Enterprise credentials in pcap, hashcat, JTR or plaintext formats
5. All without impacting out-of-scope neighbors from an easy, cross-platform web interface

This makes the Pineapple a powerful, simple and cheap option compared to any another setup required to cover all the options. Which is why this became the hardware of choice.

4.3.4 Active Bluetooth Sniffing and MiTM-attack

For active bluetooth sniffing the choice came down to Gattack as it was easy to set up and use. This tool, together with Ubertooth and Crackle was chosen in the effort of attempting MiTM-attack against bluetooth, as they would replicate a real MAC-address, sniff an existing connection and crack the LTK respectively.

4.4 Discussion

Making the choice for ease of use, POTS and portability was important for any possible future research. Any researcher should not be expected to invest heavily to be able to contribute to any field in security or regards to privacy, or any field in general for that matter.

The choices made towards software in the experiments comes down to the prerequisites of the researcher and any knowledge obtained during research. Kismet, together with Ubertooth was introduced and explained by supervisors. They had knowledge on the usability and functionality of other available software as well. Thus, making the choice of using Kismet and Ubertooth very reasonable.

The decision of choosing the Alfa adapters came down to the price and ease of use, which in hindsight proved to require drivers to be able to run. This was obviously a set back, but in return did not cause any excessive pain as it was easy to install the drivers. Hence, making the Alfa adapters most presumably a good choice.

The decision towards active bluetooth sniffing and MiTM-attacks was not an easy one. There is not much research that contains a step-by-step guide on how to perform this attack, and with what software and hardware. Therefore, making this choice very hard and the choice made could possibly have been improved with further research and greater knowledge in this field.

Chapter 5

Results

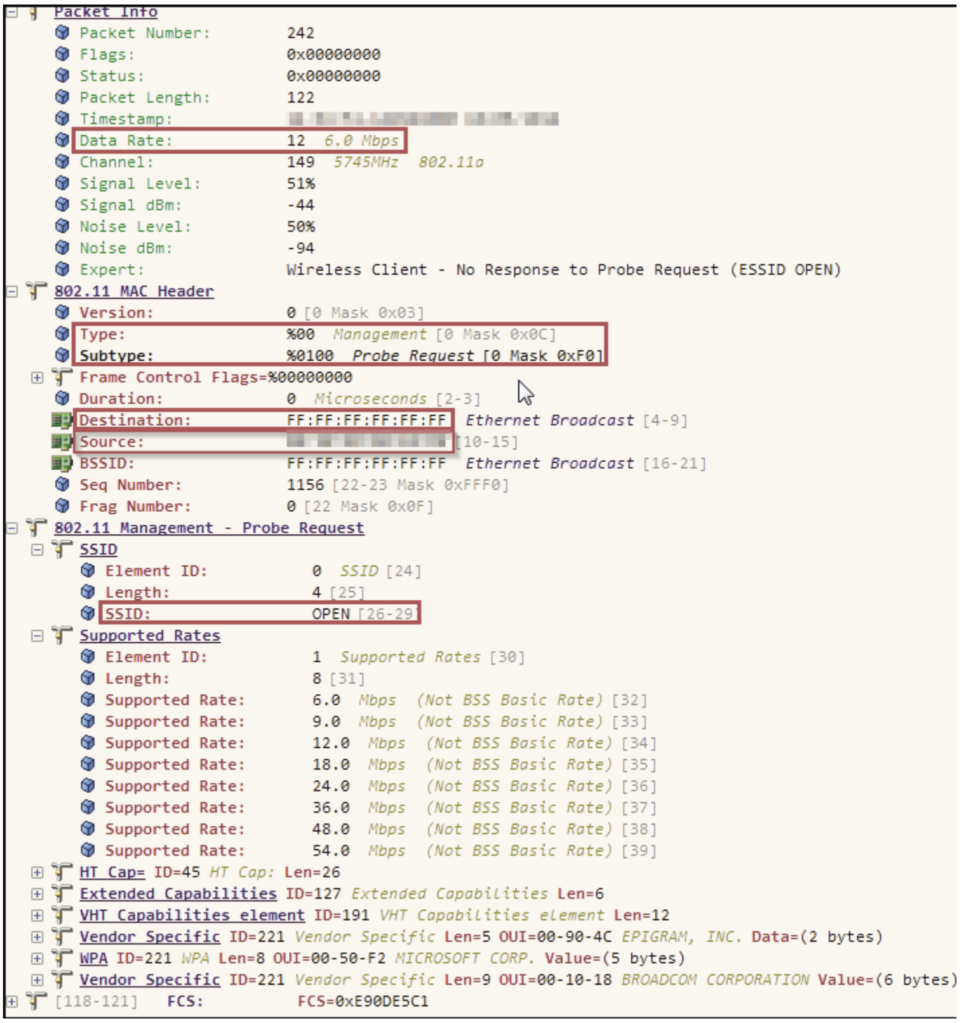
In this chapter the results from experiments are shown and explained in detail. For my experiments, I tested a *LG G5* running with Android version 8.0.0, with corresponding security update from 1st of November 2018. The *iPhone 6* tested was running iOS version 12.2 at the time of testing. During the tests, the distance between the experimental setup and the target UE varied from ~1 meter to about ~20 meters. Multiple Wi-Fi APs were set up with different security implementations for testing in this thesis.

5.1 Passive Location Tracking

The results for passive location tracking are shown and described in the following subsections. Passive location tracking is part of the *Reconnaissance* phase of the CKC, described in Table 1.2.

5.1.1 Passive Wi-Fi Sniffing

Through Wireshark or *Airodump-ng* it is possible to use monitor mode on our wireless adapter to sniff wireless packets. After the traffic is dumped, we can analyse the packets to see if we find any probe requests. In Figure 5.1 one such packet was found including its real MAC-address.

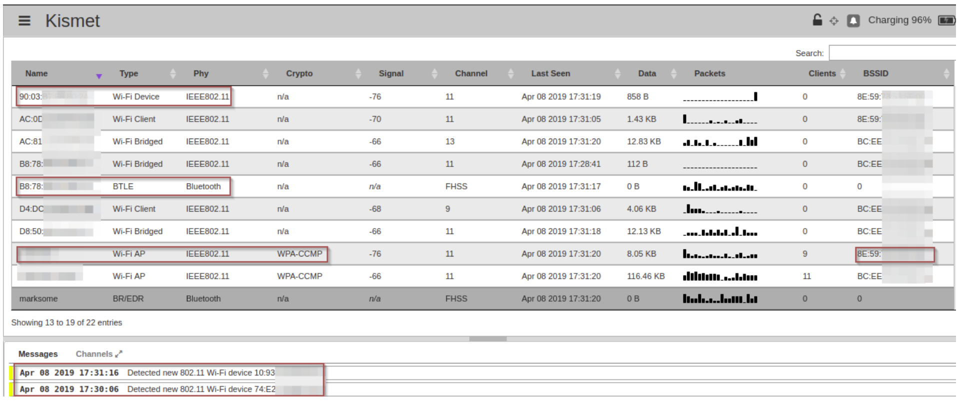


```
⊟ Packet Info
    Packet Number:          242
    Flags:                  0x00000000
    Status:                 0x00000000
    Packet Length:          122
    Timestamp:              [redacted]
    Data Rate:              12  6.0 Mbps
    Channel:                149  5745MHz  802.11a
    Signal Level:           51%
    Signal dBm:             -44
    Noise Level:            50%
    Noise dBm:              -94
    Expert:                 Wireless Client - No Response to Probe Request (ESSID OPEN)
⊟ 802.11 MAC Header
    Version:                0 [0 Mask 0x03]
    Type:                   %00  Management [0 Mask 0x0C]
    Subtype:                %0100  Probe Request [0 Mask 0xF0]
  ⊞ Frame Control Flags=%00000000
    Duration:               0  Microseconds [2-3]
    Destination:            FF:FF:FF:FF:FF:FF  Ethernet Broadcast [4-9]
    Source:                 [redacted] [10-15]
    BSSID:                  FF:FF:FF:FF:FF:FF  Ethernet Broadcast [16-21]
    Seq Number:             1156 [22-23 Mask 0xFFF0]
    Frag Number:            0 [22 Mask 0x0F]
⊟ 802.11 Management - Probe Request
  ⊟ SSID
      Element ID:             0  SSID [24]
      Length:                 4 [25]
      SSID:                   OPEN [26-29]
  ⊟ Supported Rates
      Element ID:             1  Supported Rates [30]
      Length:                 8 [31]
      Supported Rate:         6.0  Mbps  (Not BSS Basic Rate) [32]
      Supported Rate:         9.0  Mbps  (Not BSS Basic Rate) [33]
      Supported Rate:         12.0  Mbps  (Not BSS Basic Rate) [34]
      Supported Rate:         18.0  Mbps  (Not BSS Basic Rate) [35]
      Supported Rate:         24.0  Mbps  (Not BSS Basic Rate) [36]
      Supported Rate:         36.0  Mbps  (Not BSS Basic Rate) [37]
      Supported Rate:         48.0  Mbps  (Not BSS Basic Rate) [38]
      Supported Rate:         54.0  Mbps  (Not BSS Basic Rate) [39]
  ⊞ HT Cap= ID=45 HT Cap: Len=26
  ⊞ Extended Capabilities ID=127 Extended Capabilities Len=6
  ⊞ VHT Capabilities element ID=191 VHT Capabilities element Len=12
  ⊞ Vendor Specific ID=221 Vendor Specific Len=5 OUI=00-90-4C EPIGRAM, INC. Data=(2 bytes)
  ⊞ WPA ID=221 WPA Len=8 OUI=00-50-F2 MICROSOFT CORP. Value=(5 bytes)
  ⊞ Vendor Specific ID=221 Vendor Specific Len=9 OUI=00-10-18 BROADCOM CORPORATION Value=(6 bytes)
⊞ [118-121]    FCS:          FCS=0xE90DE5C1
```

Figure 5.1: Probe request including its real MAC-address.

The feasibility and limitations of this approach are that it requires a short distance of less than approximately 100 meters, and that the real MAC-address is included in the packet. Even with these conditions met, the time it takes to dump traffic and then analyse it takes too much time to be able to target specific UE.

Another approach is by using Kismet which gives information about the device manufacturer, MAC-address, the model, packets sent and received and in some cases even the name of the device in a real-time environment. All this information is sometimes leaked passively from the UE, depending on the Operating System (OS) and security version running. Kismet had the ability to add both wireless adapters and bluetooth adapters, both the bluetooth devices and wireless devices can be seen in Figure 5.2.



Kismet — Charging 96%

Search:

Name	Type	Phy	Crypto	Signal	Channel	Last Seen	Data	Packets	Clients	BSSID
90:03:[redacted]	Wi-Fi Device	IEEE802.11	n/a	-76	11	Apr 08 2019 17:31:19	858 B		0	8E:59:[redacted]
AC:0D[redacted]	Wi-Fi Client	IEEE802.11	n/a	-70	11	Apr 08 2019 17:31:05	1.43 KB		0	8E:59:[redacted]
AC:81[redacted]	Wi-Fi Bridged	IEEE802.11	n/a	-66	13	Apr 08 2019 17:31:20	12.83 KB		0	BC:EE[redacted]
B8:78:[redacted]	Wi-Fi Bridged	IEEE802.11	n/a	-66	11	Apr 08 2019 17:28:41	112 B		0	BC:EE[redacted]
B8:78:[redacted]	BTLE	Bluetooth	n/a	n/a	FHSS	Apr 08 2019 17:31:17	0 B		0	0
D4:DC[redacted]	Wi-Fi Client	IEEE802.11	n/a	-68	9	Apr 08 2019 17:31:06	4.06 KB		0	BC:EE[redacted]
D8:50:[redacted]	Wi-Fi Bridged	IEEE802.11	n/a	-66	11	Apr 08 2019 17:31:18	12.13 KB		0	BC:EE[redacted]
[redacted]	Wi-Fi AP	IEEE802.11	WPA-CCMP	-76	11	Apr 08 2019 17:31:20	8.05 KB		9	8E:59:[redacted]
[redacted]	Wi-Fi AP	IEEE802.11	WPA-CCMP	-66	11	Apr 08 2019 17:31:20	116.46 KB		11	BC:EE[redacted]
marksome	BR/EDR	Bluetooth	n/a	n/a	FHSS	Apr 08 2019 17:31:20	0 B		0	0

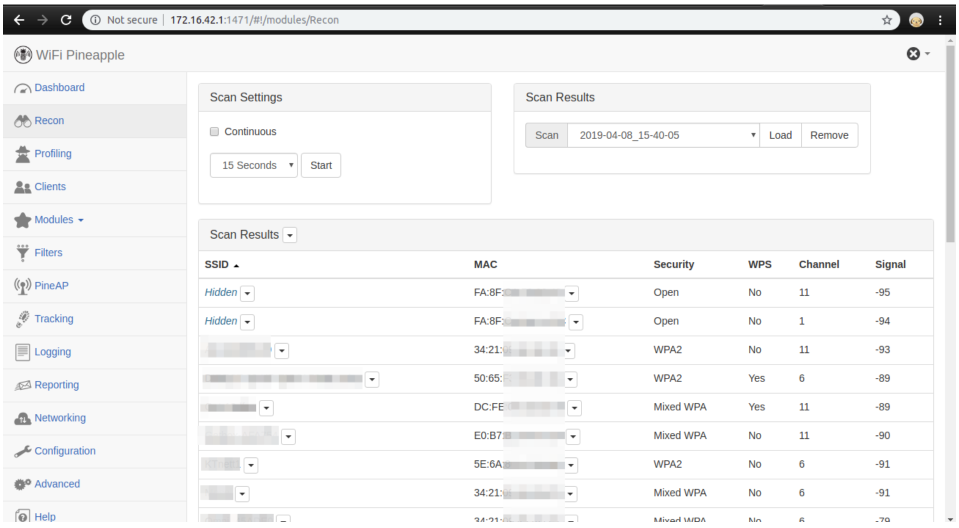
Showing 13 to 19 of 22 entries

Messages Channels

```
Apr 08 2019 17:31:16  Detected new 802.11 Wi-Fi device 10:93[redacted]
Apr 08 2019 17:30:06  Detected new 802.11 Wi-Fi device 74:E2[redacted]
```

Figure 5.2: Kismet showing nearby devices including their MAC-address.

The MAC-addresses are blurred as they are defined as personal data when collected through methods of tracking in multiple countries including Norway [2]. The limitations of this approach are that it requires a computer or Raspberry Pi to run with suitable adapters and power source. The feasibility on the other hand is very likely as the equipment would not be of size, therefore having the option of using a backpack to hide the equipment together with a power source. Furthermore, the signal strength is recorded for each packet received, hence making it possible for subspace and mesh network tracking to locate the carrier of the UE.



Scan Settings

Continuous

15 Seconds — Start

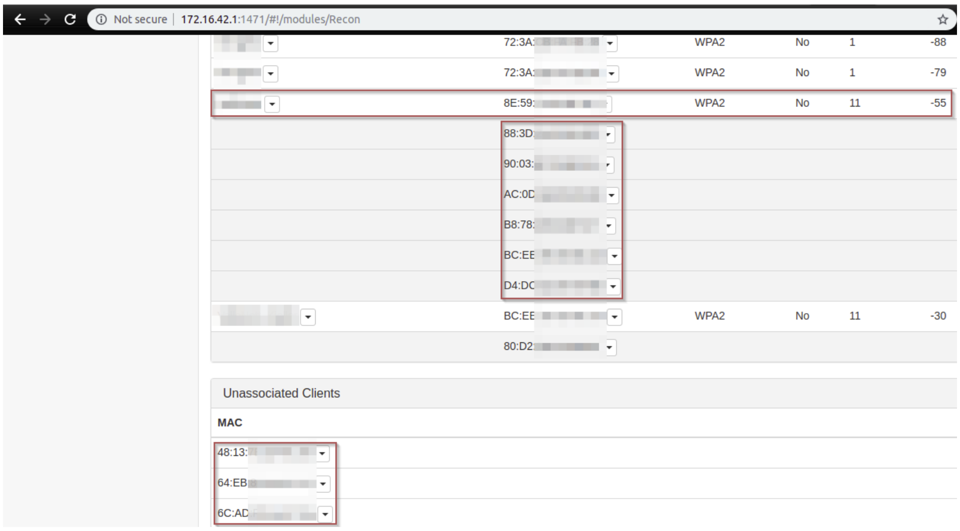
Scan Results

Scan — 2019-04-08_15-40-05 — Load — Remove

Scan Results

SSID	MAC	Security	WPS	Channel	Signal
Hidden	FA:8F:[redacted]	Open	No	11	-95
Hidden	FA:8F:[redacted]	Open	No	1	-94
[redacted]	34:21:[redacted]	WPA2	No	11	-93
[redacted]	50:65:[redacted]	WPA2	Yes	6	-89
[redacted]	DC:FE[redacted]	Mixed WPA	Yes	11	-89
[redacted]	E0:B7[redacted]	Mixed WPA	No	11	-90
[redacted]	5E:6A[redacted]	WPA2	No	6	-91
[redacted]	34:21:[redacted]	Mixed WPA	No	6	-91
[redacted]	34:21:[redacted]	Mixed WPA	No	6	-79

Figure 5.3: Pineapple recon function showing nearby devices including their MAC-address.



Not secure | 172.16.42.1:1471/#!/modules/Recon

[redacted]	72:3A:[redacted]	WPA2	No	1	-88
[redacted]	72:3A:[redacted]	WPA2	No	1	-79
[redacted]	8E:59:[redacted]	WPA2	No	11	-55
	88:3D[redacted]				
	90:03:[redacted]				
	AC:0D[redacted]				
	B8:78:[redacted]				
	BC:EE[redacted]				
	D4:DC[redacted]				
[redacted]	BC:EE[redacted]	WPA2	No	11	-30
	80:D2[redacted]				

Unassociated Clients

MAC
48:13:[redacted]
64:EB[redacted]
6C:AD[redacted]

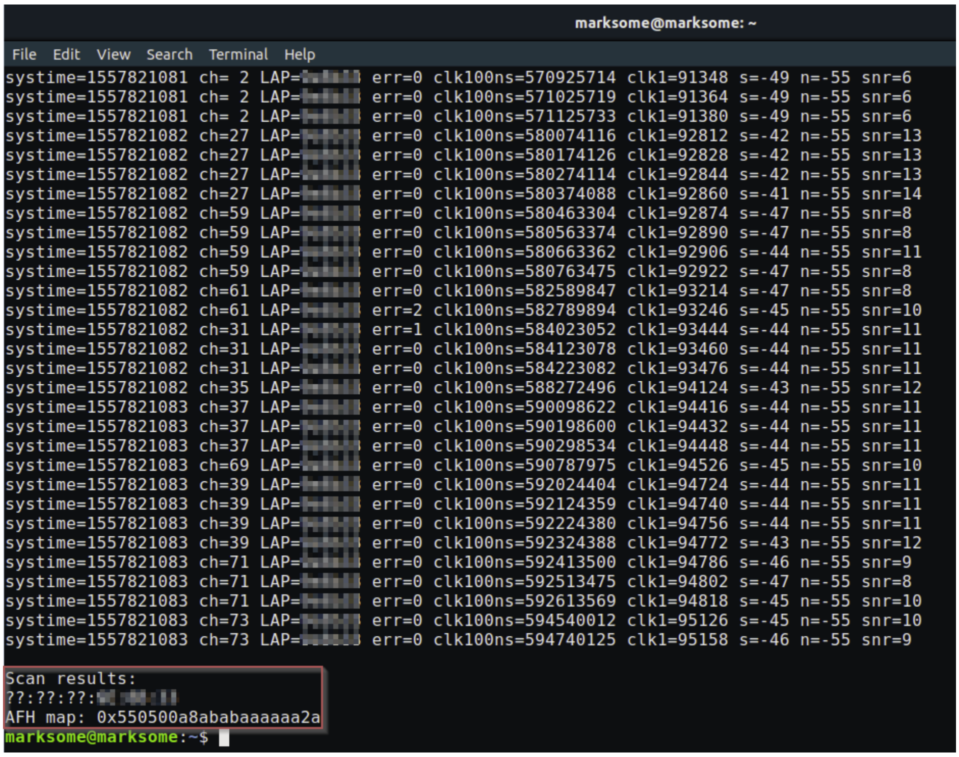
Figure 5.4: Pineapple recon function showing APs and its currently connected devices.

For passive location tracking the other option was to use Hak5's Pineapple Nano with its recon function. The recon function gives information about the SSID, BSSID or MAC-address, Security, Wi-Fi Protected Setup (WPS), Channel and Signal strength. From Figure 5.3, we can observe two hidden networks that has no security implemented. Thus, hiding the SSID in the AP configuration in an attempt to improve the privacy and security is incorrect. From Figure 5.4, we observe the currently connected devices to the different APs. Making it possible to associate devices and APs.

Apart from discovering APs, the recon function finds nearby UE probing for networks or currently connected to a network, in which case it will show which one as seen in Figure 5.4. The limitation of Pineapple is that it does not have the option of finding bluetooth devices. On the other hand, the Pineapple does show what network the UE is currently connected to, which takes a step further in regards to location tracking. The feasibility of the Pineapple is very good, Hak5 delivers a Nano version and has the option of running on your mobile. Which improves the aspect of clandestine location tracking.

5.1.2 Passive Bluetooth Sniffing

For passive tracking of active bluetooth connections, in addition to the method of using Kismet, multiple Ubertooth (*Uberteeth*), see Section 4.1.6 Ubertooth Setup), are used to sniff on the 79 designated frequency channels in regular bluetooth, whilst BLE only has 40 designated channels [22]. The Ubertooth is designed with software that can discover and follow an active connection even though the connection has AFH enabled, which usually performs 1600 channel hops per second [42]. The AFH was found by using the *ubertooth-scan*, see Figure 5.5, which gives the possibility to follow the active connection as it jumps channels. This can be utilized to obtain the MAC-address of the devices and the possibility to decrypt the connection if we manage to obtain the LTK of the connection.



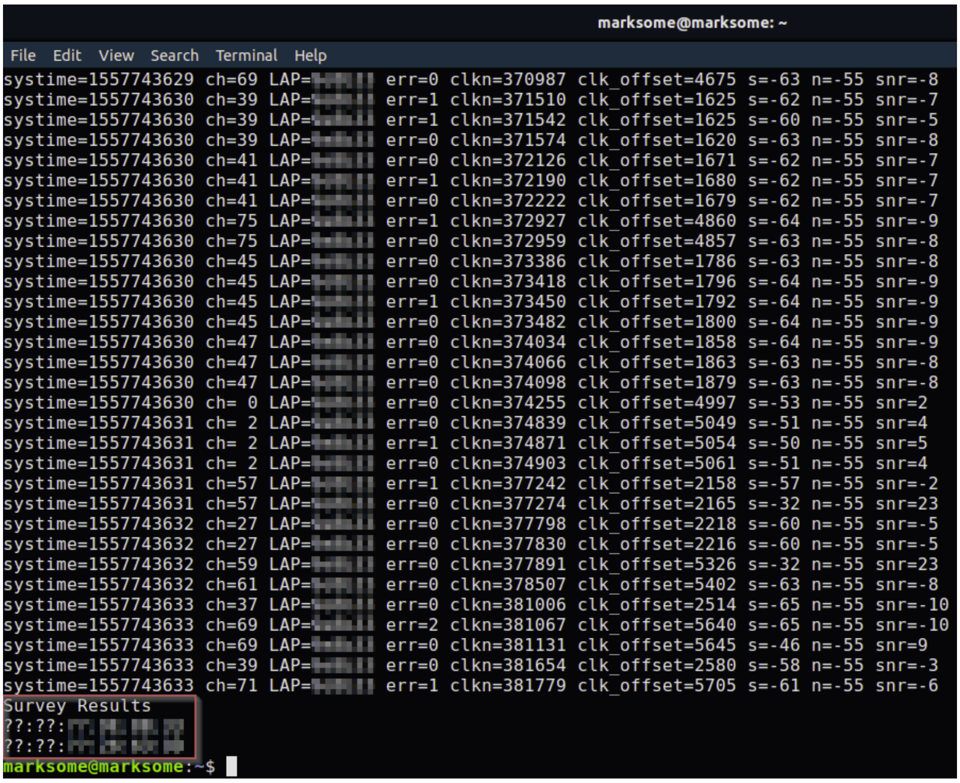
```
systime=1557821081 ch= 2 LAP=[redacted] err=0 clk100ns=570925714 clk1=91348 s=-49 n=-55 snr=6
systime=1557821081 ch= 2 LAP=[redacted] err=0 clk100ns=571025719 clk1=91364 s=-49 n=-55 snr=6
systime=1557821081 ch= 2 LAP=[redacted] err=0 clk100ns=571125733 clk1=91380 s=-49 n=-55 snr=6
systime=1557821082 ch=27 LAP=[redacted] err=0 clk100ns=580074116 clk1=92812 s=-42 n=-55 snr=13
systime=1557821082 ch=27 LAP=[redacted] err=0 clk100ns=580174126 clk1=92828 s=-42 n=-55 snr=13
systime=1557821082 ch=27 LAP=[redacted] err=0 clk100ns=580274114 clk1=92844 s=-42 n=-55 snr=13
systime=1557821082 ch=27 LAP=[redacted] err=0 clk100ns=580374088 clk1=92860 s=-41 n=-55 snr=14
systime=1557821082 ch=59 LAP=[redacted] err=0 clk100ns=580463304 clk1=92874 s=-47 n=-55 snr=8
systime=1557821082 ch=59 LAP=[redacted] err=0 clk100ns=580563374 clk1=92890 s=-47 n=-55 snr=8
systime=1557821082 ch=59 LAP=[redacted] err=0 clk100ns=580663362 clk1=92906 s=-44 n=-55 snr=11
systime=1557821082 ch=59 LAP=[redacted] err=0 clk100ns=580763475 clk1=92922 s=-47 n=-55 snr=8
systime=1557821082 ch=61 LAP=[redacted] err=0 clk100ns=582589847 clk1=93214 s=-47 n=-55 snr=8
systime=1557821082 ch=61 LAP=[redacted] err=2 clk100ns=582789894 clk1=93246 s=-45 n=-55 snr=10
systime=1557821082 ch=31 LAP=[redacted] err=1 clk100ns=584023052 clk1=93444 s=-44 n=-55 snr=11
systime=1557821082 ch=31 LAP=[redacted] err=0 clk100ns=584123078 clk1=93460 s=-44 n=-55 snr=11
systime=1557821082 ch=31 LAP=[redacted] err=0 clk100ns=584223082 clk1=93476 s=-44 n=-55 snr=11
systime=1557821082 ch=35 LAP=[redacted] err=0 clk100ns=588272496 clk1=94124 s=-43 n=-55 snr=12
systime=1557821083 ch=37 LAP=[redacted] err=0 clk100ns=590098622 clk1=94416 s=-44 n=-55 snr=11
systime=1557821083 ch=37 LAP=[redacted] err=0 clk100ns=590198600 clk1=94432 s=-44 n=-55 snr=11
systime=1557821083 ch=37 LAP=[redacted] err=0 clk100ns=590298534 clk1=94448 s=-44 n=-55 snr=11
systime=1557821083 ch=69 LAP=[redacted] err=0 clk100ns=590787975 clk1=94526 s=-45 n=-55 snr=10
systime=1557821083 ch=39 LAP=[redacted] err=0 clk100ns=592024404 clk1=94724 s=-44 n=-55 snr=11
systime=1557821083 ch=39 LAP=[redacted] err=0 clk100ns=592124359 clk1=94740 s=-44 n=-55 snr=11
systime=1557821083 ch=39 LAP=[redacted] err=0 clk100ns=592224380 clk1=94756 s=-44 n=-55 snr=11
systime=1557821083 ch=39 LAP=[redacted] err=0 clk100ns=592324388 clk1=94772 s=-43 n=-55 snr=12
systime=1557821083 ch=71 LAP=[redacted] err=0 clk100ns=592413500 clk1=94786 s=-46 n=-55 snr=9
systime=1557821083 ch=71 LAP=[redacted] err=0 clk100ns=592513475 clk1=94802 s=-47 n=-55 snr=8
systime=1557821083 ch=71 LAP=[redacted] err=0 clk100ns=592613569 clk1=94818 s=-45 n=-55 snr=10
systime=1557821083 ch=73 LAP=[redacted] err=0 clk100ns=594540012 clk1=95126 s=-45 n=-55 snr=10
systime=1557821083 ch=73 LAP=[redacted] err=0 clk100ns=594740125 clk1=95158 s=-46 n=-55 snr=9

Scan results:
??:??:??:[redacted]
AFH map: 0x550500a8ababaaaaaa2a
marksome@marksome:~$
```

Figure 5.5: Ubertooth-scan function calculating the AFH.

After the AFH is obtained we can use the following command to obtain the LAP and UAP parts of the MAC-address.

```
$ ubertooth-rx -U 0,1,2 -z -t 600
```

The command uses all three *Uberteeth* and is trying to figure out the LAP before calculating the UAP and runs for a total of 600 seconds. In Figure 5.6, the ubertooth gives information every time a packet is sniffed. The channel, in which the packet was found, and LAP is printed together with additional information that will not be focused in this thesis.



```
systime=1557743629 ch=69 LAP=[redacted] err=0 clkn=370987 clk_offset=4675 s=-63 n=-55 snr=-8
systime=1557743630 ch=39 LAP=[redacted] err=1 clkn=371510 clk_offset=1625 s=-62 n=-55 snr=-7
systime=1557743630 ch=39 LAP=[redacted] err=1 clkn=371542 clk_offset=1625 s=-60 n=-55 snr=-5
systime=1557743630 ch=39 LAP=[redacted] err=0 clkn=371574 clk_offset=1620 s=-63 n=-55 snr=-8
systime=1557743630 ch=41 LAP=[redacted] err=0 clkn=372126 clk_offset=1671 s=-62 n=-55 snr=-7
systime=1557743630 ch=41 LAP=[redacted] err=1 clkn=372190 clk_offset=1680 s=-62 n=-55 snr=-7
systime=1557743630 ch=41 LAP=[redacted] err=0 clkn=372222 clk_offset=1679 s=-62 n=-55 snr=-7
systime=1557743630 ch=75 LAP=[redacted] err=1 clkn=372927 clk_offset=4860 s=-64 n=-55 snr=-9
systime=1557743630 ch=75 LAP=[redacted] err=0 clkn=372959 clk_offset=4857 s=-63 n=-55 snr=-8
systime=1557743630 ch=45 LAP=[redacted] err=0 clkn=373386 clk_offset=1786 s=-63 n=-55 snr=-8
systime=1557743630 ch=45 LAP=[redacted] err=0 clkn=373418 clk_offset=1796 s=-64 n=-55 snr=-9
systime=1557743630 ch=45 LAP=[redacted] err=1 clkn=373450 clk_offset=1792 s=-64 n=-55 snr=-9
systime=1557743630 ch=45 LAP=[redacted] err=0 clkn=373482 clk_offset=1800 s=-64 n=-55 snr=-9
systime=1557743630 ch=47 LAP=[redacted] err=0 clkn=374034 clk_offset=1858 s=-64 n=-55 snr=-9
systime=1557743630 ch=47 LAP=[redacted] err=0 clkn=374066 clk_offset=1863 s=-63 n=-55 snr=-8
systime=1557743630 ch=47 LAP=[redacted] err=0 clkn=374098 clk_offset=1879 s=-63 n=-55 snr=-8
systime=1557743630 ch= 0 LAP=[redacted] err=0 clkn=374255 clk_offset=4997 s=-53 n=-55 snr=2
systime=1557743631 ch= 2 LAP=[redacted] err=0 clkn=374839 clk_offset=5049 s=-51 n=-55 snr=4
systime=1557743631 ch= 2 LAP=[redacted] err=1 clkn=374871 clk_offset=5054 s=-50 n=-55 snr=5
systime=1557743631 ch= 2 LAP=[redacted] err=0 clkn=374903 clk_offset=5061 s=-51 n=-55 snr=4
systime=1557743631 ch=57 LAP=[redacted] err=1 clkn=377242 clk_offset=2158 s=-57 n=-55 snr=-2
systime=1557743631 ch=57 LAP=[redacted] err=0 clkn=377274 clk_offset=2165 s=-32 n=-55 snr=23
systime=1557743632 ch=27 LAP=[redacted] err=0 clkn=377798 clk_offset=2218 s=-60 n=-55 snr=-5
systime=1557743632 ch=27 LAP=[redacted] err=0 clkn=377830 clk_offset=2216 s=-60 n=-55 snr=-5
systime=1557743632 ch=59 LAP=[redacted] err=0 clkn=377891 clk_offset=5326 s=-32 n=-55 snr=23
systime=1557743632 ch=61 LAP=[redacted] err=0 clkn=378507 clk_offset=5402 s=-63 n=-55 snr=-8
systime=1557743633 ch=37 LAP=[redacted] err=0 clkn=381006 clk_offset=2514 s=-65 n=-55 snr=-10
systime=1557743633 ch=69 LAP=[redacted] err=2 clkn=381067 clk_offset=5640 s=-65 n=-55 snr=-10
systime=1557743633 ch=69 LAP=[redacted] err=0 clkn=381131 clk_offset=5645 s=-46 n=-55 snr=9
systime=1557743633 ch=39 LAP=[redacted] err=0 clkn=381654 clk_offset=2580 s=-58 n=-55 snr=-3
systime=1557743633 ch=71 LAP=[redacted] err=1 clkn=381779 clk_offset=5705 s=-61 n=-55 snr=-6
Survey Results
??:??:[redacted]
??:??:[redacted]
marksome@marksome:~$
```

Figure 5.6: Ubertooth passive sniffing for LAP and UAP of MAC-address.

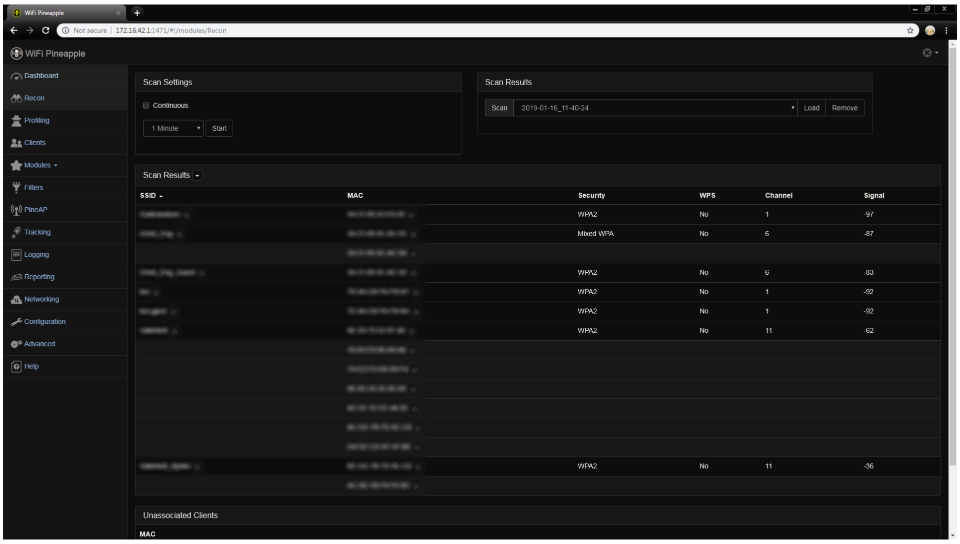
The limitations of this approach are that the target UE has to be close to our experimental setup and for a longer period of time, about 600 seconds or more to be reliable. The feasibility is then limited, but possible if the target UE and carrier is known and can be stalked.

5.2 Active Location Tracking

There are multiple possibilities when it comes to active location tracking and are described in the following subsections. Active location tracking would be part of the *Delivery* and *Exploitation* in the CKC, described in Table 1.2.

5.2.1 Active Wi-Fi Sniffing

When performing active Wi-Fi sniffing, the option of having a Hak5 Pineapple was great. The results were far better than what was initially thought. In Figure 5.7 it is not easy to see, but the recon function listed MAC-addresses from UE and their currently connected AP, and the MAC-address of the AP.



Scan Settings

Continuous

1 Minute

Scan Results

2019-01-16_11-40-24

Scan Results

SSID	MAC	Security	WPS	Channel	Signal
[redacted]	[redacted]	WPA2	No	1	-97
[redacted]	[redacted]	Mixed WPA	No	6	-87
	[redacted]				
[redacted]	[redacted]	WPA2	No	6	-83
[redacted]	[redacted]	WPA2	No	1	-92
[redacted]	[redacted]	WPA2	No	1	-92
[redacted]	[redacted]	WPA2	No	11	-62
	[redacted]				
	[redacted]				
	[redacted]				
	[redacted]				
	[redacted]				
	[redacted]				
[redacted]	[redacted]	WPA2	No	11	-36
	[redacted]				

Unassociated Clients

MAC

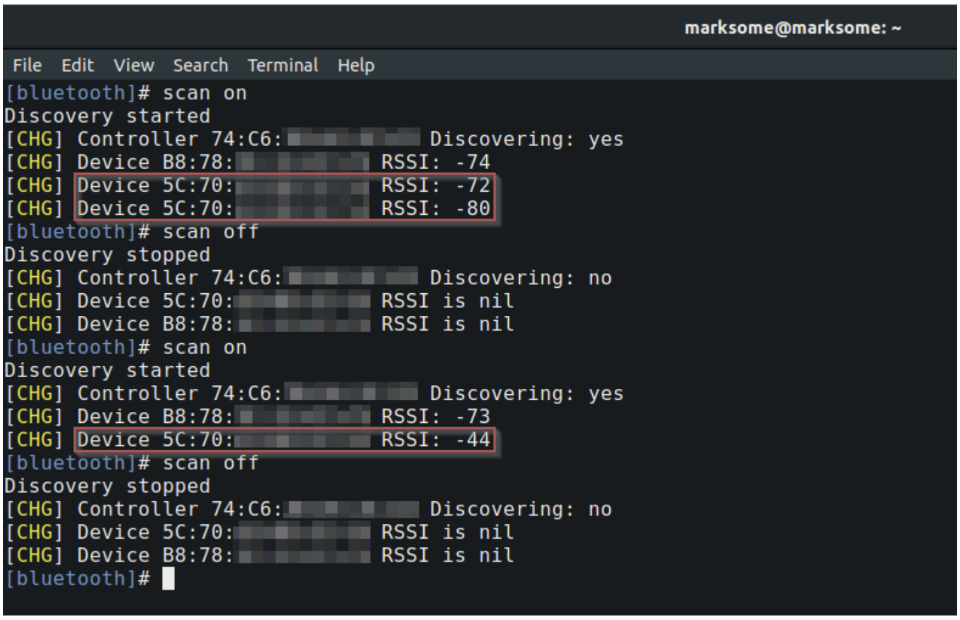
Figure 5.7: Pineapple using Recon function.

While sniffing for probe requests the Pineapple can send out broadcast frames as well as replicate common SSIDs such as hotels, airplanes and café hotspots, to retrieve more requests than during passive sniffing. The difference is that the UE include their real MAC-address when attempting to connect to our Pineapple. This is in some cases sent in the connection request, and in some cases during the handshake between the two devices.

A limitation to this method is if the UE has implemented randomization of the MAC-address, for each new connection, making the feasibility less likely for location tracking. Furthermore, most equipment doesn't have this implementation active, hence making the feasibility most likely and more so compared to clandestine location tracking. This is because more often than not, the real MAC-address will be sniffed from the wireless traffic when performing active sniffing.

5.2.2 Active Bluetooth Sniffing

During active bluetooth sniffing, our bluetooth adapter was set into pairing mode to find and recognise other bluetooth devices in the vicinity of our experimental setup. Thus, the available bluetooth devices would be easily listed, and we managed to attempt connecting to these devices to obtain their MAC-address. For this experiment we had a simple setup with the two phones, where one of them was moving and the other stationary, see Figure 5.8.



```
[bluetooth]# scan on
Discovery started
[CHG] Controller 74:C6:[redacted] Discovering: yes
[CHG] Device B8:78:[redacted] RSSI: -74
[CHG] Device 5C:70:[redacted] RSSI: -72
[CHG] Device 5C:70:[redacted] RSSI: -80
[bluetooth]# scan off
Discovery stopped
[CHG] Controller 74:C6:[redacted] Discovering: no
[CHG] Device 5C:70:[redacted] RSSI is nil
[CHG] Device B8:78:[redacted] RSSI is nil
[bluetooth]# scan on
Discovery started
[CHG] Controller 74:C6:[redacted] Discovering: yes
[CHG] Device B8:78:[redacted] RSSI: -73
[CHG] Device 5C:70:[redacted] RSSI: -44
[bluetooth]# scan off
Discovery stopped
[CHG] Controller 74:C6:[redacted] Discovering: no
[CHG] Device 5C:70:[redacted] RSSI is nil
[CHG] Device B8:78:[redacted] RSSI is nil
[bluetooth]# 
```

Figure 5.8: Bluetoothctl using scan function.

This is very feasible as it can be conducted with any bluetooth capable device, but has the high limitation of devices needing to be set into pairing mode in order to be discovered. Therefore, making this option not reliable in an effort to track devices, but a combination of multiple methods could return better results.

5.3 Eavesdropping

Eavesdropping is the act of gaining information from an active connection without the UE knowing.

5.3.1 Wireless MiTM Attack

Wireless MiTM-attacks are known as the Evil Twin attack. This is performed by getting in between a real connection. During experiments, the setup had a *LG G5* phone that was connected to the office network through WPA2. The first part included obtaining the PSK, which was retrieved by monitoring all connections to the AP before sending an *arp-deauth* attack to the AP. This would make the all devices reconnect to the AP, hence including the PSK which we will sniff from our monitoring. The PSK is encrypted at this stage, but by using *Aircrack* the PSK can be calculated using a rainbow table or a list of most common PSKs. After the PSK is cracked, the Pineapple will replicate the BSSID of the target AP and send *arp-deauths* until the target UE connects to the Pineapple, which again is connected to the target AP. This will make both devices unaware of our presence, making it possible to sniff the unencrypted traffic. Possibilities include attempting a downgrade attack, to try to enforce the use of HTTP and not HTTPS, or inject javascript code, to make the device act on our behalf or download malware. The objective of the thesis is eavesdropping and not injecting, hence making MiTM-injection not relevant for this thesis, but is recommended for future work.

5.3.2 Bluetooth MiTM Attack

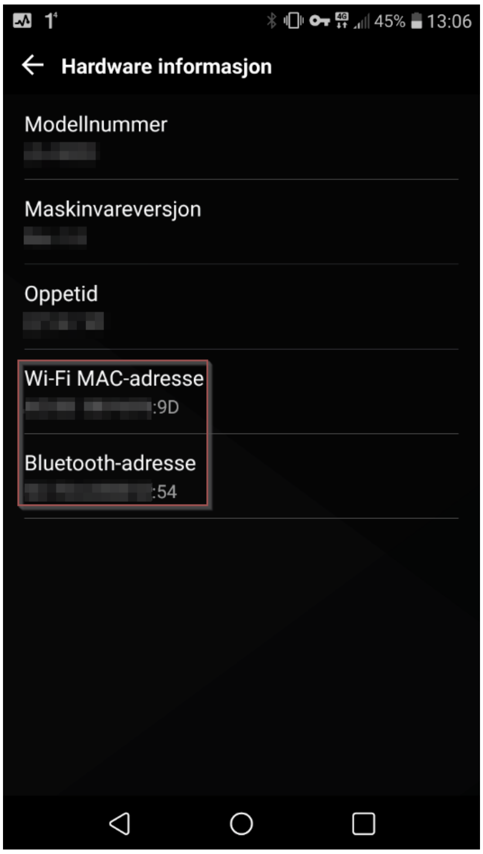
During the experiments, there was a lot of time invested in attempting MiTM attacks on the bluetooth protocol. After multiple attempts with multiple different setups with both hardware and software, the knowledge obtained during research did not add up for a successful attack to be made. The main goal was to first retrieve the MAC-address of both connected devices. The second objective was to spoof the MAC-address of device number one and thirdly jam the signal from device number two. After this was done, we could intervene and connect our own device to the desired target. This would have to include the LTK or a downgrade attack, where the security level is set to 1 and security mode 1 is used.

5.4 Correlation Between Wi-Fi and Bluetooth

During the literature study it was made the discovery of a possible correlation between the MAC-addresses of the Wi-Fi and bluetooth hardware. The bluetooth address had, in some cases, a correlation from the Wi-Fi address where it had a one bit higher value than the Wi-Fi address, corresponding to the last byte.

5.4.1 Correlation on Android 8.0.0

As seen from Figure 5.9, this was not the case as of security version 1st of November 2018, but was in fact the case before this security update. This makes it not possible to correlate these two addresses, which in reality makes clandestine or active tracking in some ways harder.


Hardware informasjon

Modellnummer
[redacted]

Maskinvareversjon
[redacted]

Oppetid
[redacted]

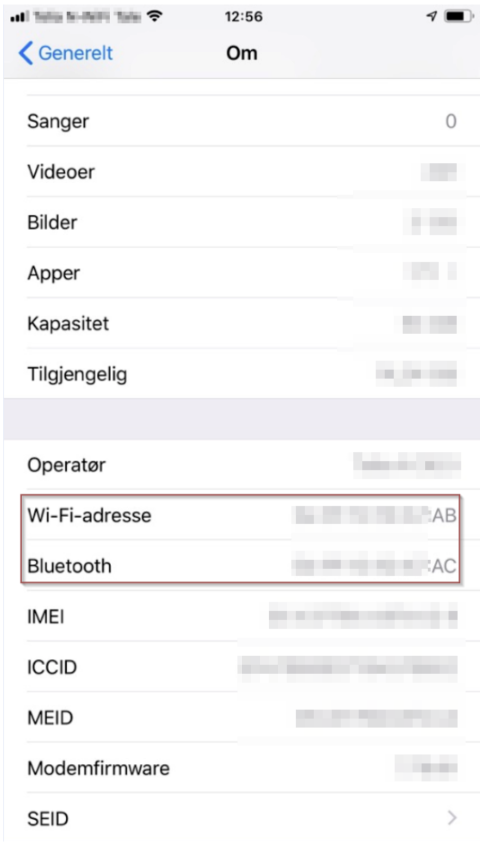
Wi-Fi MAC-adresse
[redacted]:9D

Bluetooth-adresse
[redacted]:54

Figure 5.9: Correlation between Wi-Fi and bluetooth on Android version 8.0.0.

5.4.2 Correlation on iOS 12.3.1

As we see in Figure 5.10, correlation is actually the case for iOS version 12.3.1. Which makes a Wi-Fi address of example *aa:aa:aa:aa:aa:aa* to become the bluetooth address of *aa:aa:aa:aa:aa:ab*. Thus, making tracking of either bluetooth or Wi-Fi on iOS version 12.3.1 and earlier an easier task, as you could find the UE, such as mobile terminals, by either its Wi-Fi or Bluetooth address.



[redacted] 12:56

Generelt **Om**

Sanger	0
Videoer	[redacted]
Bilder	[redacted]
Apper	[redacted]
Kapasitet	[redacted]
Tilgjengelig	[redacted]
Operatør	[redacted]
Wi-Fi-adresse	[redacted]:AB
Bluetooth	[redacted]:AC
IMEI	[redacted]
ICCID	[redacted]
MEID	[redacted]
Modemfirmware	[redacted]
SEID	>

Figure 5.10: Correlation between Wi-Fi and bluetooth on iOS version 12.3.1.

5.5 Discussion

For passive attacks on both Wi-Fi and Bluetooth the objectives have been proven to be very feasible, but have some limitations. The chosen methods have proven to be successful and easy to conduct, making these methods powerful if, and only if, the real MAC-address is obtained. Therefore, the randomization of MAC-addresses creates a limitation on passive attacks.

During active attacks, the wireless setup proved to be more time consuming as the attack required more planning and a more advanced setup. While testing for common SSIDs, the success rate varied as the UE had different open hotspots (APs) stored. The most successful ones where common hotspots found in hotels. On the other side the Pineapple proved to be easy to use with its simple GUI, ease-of-use and portability. While using the recon function of the Pineapple, we obtained the MAC-address of all the devices in the vicinity including their currently connected AP. That the Pineapple listed the currently connected AP of any UE was a new discovery for the author as the connection should be encrypted, but apparently not the header in the data link packet in layer two of the Open Systems Interconnection Basic Reference Model (OSI BRM). Thus, active attacks on Wi-Fi have a high feasibility, but still include the same limitations as with passive attacks. Even though in some scenarios the UE sends its real MAC-address when connecting to our fake hotspots.

The active bluetooth sniffing was very easy to setup and conduct, but did not return the same type of results as the passive sniffing. This is because the passive sniffing returns all bluetooth devices that are turned on or connected to other devices, while the active sniffing only returned those in pairing mode. Hence, active bluetooth sniffing is highly feasible, but has the limitation of only discovering devices in pairing mode.

An observation made during testing was that some Android and iPhone devices had a correlation between the Wi-Fi MAC-address and the bluetooth MAC-address. The bluetooth address could be calculated from the Wi-Fi MAC-address by adding one bit to the last byte, and vice versa. Hence, location tracking on these devices is easier and need only one of the MAC-addresses, either from Wi-Fi or bluetooth. The observation of probe requests including the SSID from Apples iPhone was made. This is most probably the result of an owner of the UE has not updated the OS, which could be the case of more than just this phone alone. Mobile devices can run old versions of their corresponding OS, which makes clandestine tracking even more successful. This could be the case for jailbroken iPhones, rooted android devices or old UE with outdated software.

For the wireless MiTM attacks, the author has conducted similar tests, which made it easier to understand and setup, but this attack was highly more sophisticated. The attack proved to be very time consuming as the setup requires multiple factors and prerequisites to be successful. Amongst some of the prerequisites is that the signal from the Pineapple or experimental setup needs to be stronger than the target AP from the UE. This makes it harder to conduct in reality than it is in an experiment, where you can go close to the UE as you know its location. Hence the feasibility is less likely and comes with higher limitations regarding the signal strength and distance to both the target UE and the AP.

Bluetooth MiTM attacks proved to be much more time consuming and frustrating than initially planned. Multiple attempts were made to jump in-between an active connection without much success. This attack had many prerequisites that had to go our way to be successful, as we needed to jam the signal from one of the devices and try to connect to the other device almost at the same time. Without having the LTK this would prove to be next to impossible even with an attempt to downgrade the security level to discard encryption. Bluetooth MiTM would be recommended for future work, as the knowledge and research from the author of this thesis did not prove good enough.

A comparison between the different attack methods can be seen in Table 5.1.

Table 5.1: Comparison between the different attack methods.

Method	Description	Pros	Cons
Kismet	Passive sniffing	High feasibility of passive sniffing on both protocols	Limitation that obtained MAC could be randomized
Pineapple	Multipurpose tool	High feasibility with its portability to conduct passive, active sniffing and MiTM on Wi-Fi	Medium limitations as active sniffing and MiTM depends on multiple factors
Ubertooth	Passive sniffing	High feasibility to sniff on active connections	Limitation that obtained MAC could be randomized
Bluetooth scan	Bluetooth device in pairing mode	High feasibility to discover active devices in pairing mode	Most devices are not in pairing mode
Wi-Fi MiTM	MiTM between UE and AP	Potential to compromise privacy and security of device	Low feasibility and high limitations
Bluetooth MiTM	MiTM between two devices	Potential to compromise privacy and security of device	Very low feasibility and very high limitations

Kismet is the most effective method when it comes to passive sniffing, with its high feasibility and usability with both protocols. The Pineapple is probably the most effective method covering three objectives of the Wi-Fi protocol. Its feasibility is very high with its simplicity and portability, even though the limitations discussed earlier still exist. Correlating the Wi-Fi and Bluetooth address could be a potential threat to privacy, depending on whether randomization is applied or not. Therefore, a combination of Pineapple, Kismet and correlating of addresses would be a great threat to privacy, in regards to clandestine and active location tracking.

Chapter 6

Countermeasures

In this chapter possible countermeasures are discussed, to mitigate the attacks and results described in Chapter 5 Results, and possible future attacks on the protocols preventing active and clandestine location tracking as well as MiTM-attacks.

6.1 Anonymous Probe Requests

We learned from Section 5.1 Passive Location Tracking the MAC-address is sometimes included in the MAC-header as we see in Figure 5.1. A suggested solution would be anonymous probe requests. This could be implemented by making all devices use the same probe when asking for WLANs. In Figure 6.1 the *Source Address* is set to *Unknown (fa:00:00:00:00:00)*, which could be a solution to stop include the real or randomized MAC-address in the MAC-header. Thus, making any attempts of breaking the randomization seed, as attempted by Vanhoef et al [43], useless as neither the real or randomized MAC-address would be included.

```
Type/Subtype: Probe Request (0x04)
Frame Control: 0x0040 (Normal)
Duration: 0 microseconds
Destination Address: Broadcast (ff:ff:ff:ff:ff:ff)
Source Address: Unknown (fa:00:00:00:00:00)
BSS ID: Broadcast (ff:ff:ff:ff:ff:ff)
Fragment Number: 0
Sequence Number: 1
SSID Parameter Set: Broadcast
Supported Rates: 1.0 2.0 5.5 11.0 6.0 9.0 12.0 18.0
Extended Supported Rates: 24.0 36.0 48.0 54.0
DS Parameter Set: Current Channel: 1
```

Figure 6.1: Probe broadcast request as it would be viewed in Wireshark.

A possible limitation of this implementation would be in the case of multiple probe requests transmitted at the same time, as the MAC-address is used for routing the wireless packets. In reality this would not be a problem as all probing devices are seeking for nearby APs, and would all receive the probe responses.

6.2 Disabling Directed Probe Requests

With directed probe requests the SSID or BSSID will be included. As discussed in Section 2.3 Wi-Fi Probe Sniffing, a search engine like *wigle* could help identify popular locations of the UE. Thus, making it critical to disable directed probe requests even for older versions of iOS and android. The result would be a higher increase in time between the first probe and the connection phase, but the compromise will be well worth it as it increases the privacy of the UE.

6.3 Randomization of MAC-address

A suggested solution would be to implement total randomization of the MAC-address for each new connection made. This suggestion would solve the problem of tracking and storing of MAC-addresses, which would compromise the privacy of the UE.

The only problem when it comes to randomization is that without a proper randomizing seed the randomization can be broken, as proven multiple times in different studies, including the studies from 2016 [43] and 2017 [44]. Therefore, it is recommended to randomize the entire six byte or twelve hexadecimal address, making it to about 281 trillion possible addresses.

This would also help remove the correlation found between the Wi-Fi- and bluetooth address on Apples iOS versions 12.3.1 and earlier. Thus, the possibility of linking these addresses in the future would not exist. This solution would also mitigate active Wi-Fi sniffing leaking the real MAC-address. The limitation of this implementation is how to create a strong randomization seed, as this would require stronger methods that what is broken by the studies from 2016 [43] and 2017 [44].

6.4 Certificates and Public/Private-Key Encryption

Certificates are great to both enhance security as well as ensuring both connected devices, both the Hotspot (Wi-Fi) and the UE. Certificates can be signed, hence making any MiTM-attacks on both Wi-Fi and Bluetooth harder in case the public certificate needs approval from any vendor or Certificate Authority (CA)-authority. The same idea mitigates attacks where the MAC-address is duplicated. In reality the certificates could potentially eliminate the need of a MAC-address, as it is only used as a way of separating devices on LANs.

When it comes to complications using certificates, one such problem would be speed. The reason for this is that certificates, for the most part, use asymmetrical Rivest–Shamir–Adleman (RSA)-encryption which in reality is time consuming compared to symmetric ciphers such as Data Encryption Standard (DES) and Advanced Encryption Standard (AES). In Bluetooth and BLE, the speed is a factor as the AFH would have complications reaching 1600 channel hops per second.

The solution to fix these problems would be to implement the initiating of the connection with certificates, then adapt into the use of any preferred security measures wanted or needed for that specific connection. This will ensure that the MAC-address will remain secure and complicate the process of tracking and in result increase the privacy of any UE.

If the less secure methods of encryption persist in Bluetooth and BLE, the attack vector shown with passive Ubertooth tracking (4.1.6 Ubertooth Setup and 5.1 Passive Location Tracking) will still prevail to influence the privacy, hence making the implementation of certificates less efficient as it will have little to no effect.

From Figure 6.2 and Figure 6.3, we can see two certificates, one public and one private respectively. These certificates could be utilized for enhanced security and privacy when encrypting both the Wi-Fi- and bluetooth handshake.

```
---- BEGIN 802.11 PUBLIC KEY ----
Comment: Example public key for Masters Thesis
AAAAB3NzaC1yc2EAAAABJQAAAgEAmkq0yOEmvlzH9p1rr/8Iyse9nOBQlFqtqc/i
Gg5SCzFB7n4tJYj/Ykr5HmIs9RdsXHOZeptUuL3Zh2Jy5lyeaBEGYxALp6H0zRO8
Uvi6Y6hGLU0/nq9bgNuh3sBOnQpH8u+8hs/seO9769ai/h6WeLbDMRK6LxKu/tY0
/16Zd6vgZQ0/E6pqRGvAK97VdDc1FvvRyRyTOK4fewpUMj7Ha+drS3NOIacjlMkn
VvHSRRZpeKG6FTDx8GVzKa2RMvgEJaNtm9qeFwv0ZOcd+NfxPgJj4bTCqrC9DCHO
B/1ooJAvbdsBMA15kSKEGrw3crji+rfpO3YJIaIv2TOr2XoUqPwcEEzQJZYt4D2V
HcDmRO11L6WJ426l2EpYurLUh7xsaZKuQq0LIwteYFHovSh7ySwX1d2P2GuwjRMK
AZQzRMKZ24XvTos2ynviw5A8Zxh+64xGjNSEzQ7FFBg6q5B3vl2VQrFqkKp7ILy6
PWJA7bRcQcXGkCY49aW+Btolam5K1OM6Mwhm3gIMS55WVNKtz25QRjqR1IZe6xy7
E/YlHRyf7Cka2/s7MUwu2mGqzatv9kkGXRi6zXbS7thx0/UbdvDBJLR6dI1AH8ui
3hw6ZN3hV+JdTwNbEua3ZyQuTMkoFur/X3s0w5YGrKHPhBEC8GSuTPklaEMMe0pe
UOBxIPs=
---- END 802.11 PUBLIC KEY ----
```

Figure 6.2: Public 802.11-certificate.

```
---- BEGIN 802.11 PRIVATE KEY ----
Comment: Example private key for Masters Thesis
AAACAHCXYVRRRcku8si4EE//S5rkyKSkjdQK0cEbKHPg78L6bmHtXzcQ8bZuD7w5
zcebeJZ93rNqbkGfS7XSOCuPsd082yWxw1DQIVdhbcABnMVDcXsBBOlrNO9UKgNp
vNN2NIDYdNEURMbz8qUt7I/szqsdlVspV251avFQNIL/18YHzULSUJ+l3tf7mhIs
/KDqBCWHU+XIvnV4JM9oZvQ7pkfSYw2LfjQ7si5NmT9yMagJb49hQpnXm8sZmToW
35Plas9blSyKxmPYSo7SKqDU7k/YOwx8/MG/PdhrsgXUmHUq8i2R+fKiDJpXdShE
KHZPkOB4XhaiPgPJYS/LwpfHyeHtl3lyE8o9cgGz/TXBjAuf6FrWtuLneD3JaRyt
BgEVMGwWpODQrr9PCCSJvvS/fet1+RbnZrvXUFQ+qpf6UCHgw0ctTr3uQKDjtU4o
JGdXZqAplGcaReeVvjuvPllguUKS0GiqSninM+f4cLKe5jRNW1vo+qqWp2kf9mXL
/hckSVPlmCjzFs4IrqLO2Yh++s1AH5054yHLGKQDyoZX1iYUs8adlxdSjHNjPPRZ
UDl3OzpVFetXmv6yWBECPku2ibMepz/XpUhW/oYawM9Wb7vDL6V2ZOIJMis1sB3J
48yavHzL02roByui2qgJ6H9r7Cq1L/LYRH7yN7VE8g0A0bK1AAABAQD/dNhaRb8b
sW7hSTME6KkCaht2Vz47pL8yvRQoq3vyrZoJcMWX/1qZObviK1KJwa8edCUFR2wk
BKgY2wBmiaMeMheT0AEpHUrqFHPoltyip2NwX1Xg3c8PdAMlx6sgQl5gCyOZprXj
tbh7bK46D7ik5C1NFN2HvpsENvgVVg7L9DwlKNSAYILzNKaSYW3N95El0C+GTRrP
bziryFZXWk8U3sIKC7QVfpq4wuuHVMdl9pAbRnGAtaPSJhoAIDJzb7olIGa+3WsC
V8oRQdvE87y4/AlKSWT6So7AU6f9g6mNdtwtO61zy9rjHkzk0V4Kwz6LAFIdk8HR
LguXUWrePkh/AAABAQCansDt6oo5Pwa6+mguBM5bwy9nxj7ZXFKkyTzfbKt9TQwB
6cRC7Og1P2jiYmcTkkkpmjKkqK09DmgMmo+6MlbFuvocVJT1rfhsuWlKtgZTZ1dL
0ErZIUVhhUPl2zCfzzQq3k1K2Q+C05wlvJphZ4UXPUEv8eVLSRZsPuIVj5q8dtnr
vBzvDQfTiYORrltgLyUDlyLNPOC9EEzwfmRV1L/FdXjCXIytgNwB8noUF36iFUWm
MBrdmQ78IS9QgQDc3BMv1/4s6fJDGcPhka1SmpPgh5HGbTkF8bE7oc837bpBuGnM
X5rJcgu3SvS45Xdo351AM2I38WpCYrAupME59wmFAAABAD90msQ+8Utf2hvLAEzH
AK+trjWnS9+Fh579xiljnIwWJyFdjfYOQPlwv80OG50GnNzRMs/qLjwgtv3ht0ag
tb+j2yOF4pbRi24DnwzBXz5z5ZUvZQjpFP3t1vhPIRVaJUcwwsBiwIMUrX277Qzk
whLBx27M+rQxUdd+njJxfo9Es5+z7vc7in73yONubMjXk7Kp9WbP3trz+w0w9DHb
6nkmsDdHatbBJKGB51E55avcAk83DZMdJEz1CcZIEjGlYtBgECiemcHs+TQj4Mve
e4HuUncQcEpxMxRAB0Yi3dwDnW2BdZJcGaY3elNdy9V6xr/ODZCz5gmBTChyB9pG
BLA=
---- END 802.11 PRIVATE KEY ----
```

Figure 6.3: Private 802.11-certificate.

The implementation of certificates would require the vendors to create unique certificates signed by a CA for each device. This would not be difficult to implement as it would only require a software patch where each device can create a unique seed from its International Mobile Equipment Identity (IMEI) or similar unique ID.

This would open the possibility to revoke certificates, even though it would require an online connection to check the legitimacy of the certificate, in case a device has been used for malicious activity. The great thing about certificates is that it would remove the possibility of MiTM. This is because the devices would either store the public certificates of the known APs, or devices in Bluetooth, or check the legitimacy of the certificates online.

Other limitations of certificates would include the routing on layer two of the

OSI BRM, because the MAC-address would no longer be open and unencrypted in the connection initiation. The idea is for the connection initiation to start in the application layer, hence the packets does not need to go through routing before the connection is established. After the connection is established, routing will work as intended and the randomized MAC-address can be again utilized in layer two. This implementation would require a huge change in how the connection initiation is set up, as it would be encrypted with the use of public certificates. This would not be backwards compatible with old equipment as this implementation would create a new security mode.

Lastly the implementation of certificates would remove the possibility of both clandestine and active location tracking. This is because the device would encrypt its MAC-address with the known or legitimate public certificate of the AP. Encrypted in the Application Layer (Layer seven in the OSI BRM), making any effort of obtaining information from the Data Link Layer (Layer two in the OSI BRM) useless. Hence, we have obtained enhanced privacy before initiating the connection handshake. How this would be implemented must be researched in future work.

6.5 Discussion

When it comes to choosing a countermeasure, there are little to no precautions that can be done by any user as the problem persists with the protocols. Any user may turn of his Wi-Fi and bluetooth when traveling, but this limits the users of using any open networks or listening to their wireless headphones. Even those connecting their phones to their cars, for hands-free speech or to play music, have a risk of getting tracked with today's standards. Thus, any mitigation's done by the user would result in limited use the protocols. This is of course not in any way optimal and is therefore not implemented as a countermeasure.

Of the mitigation's listed and suggested above, the implementation of randomization and certificates in pairing or encryption is recommended. This is because of the simplification of securing the TK, LTK and lastly the MAC-address for enhanced privacy. The certificates would be a way of assuring the equipment is legitimate as the certificates can be signed by any CA-authority, making the *Evil Twin Attack* or AP-replicating next to impossible.

In WPA-Enterprise there is an option of choosing certificates as a security implementation. This would be a great implementation if the certificates where used in the four way handshake and authentication process.

This option has not been implemented in the Bluetooth standard and would greatly improve the security as well as the privacy of the protocol. Bluetooth devices need to use Pairing-mode or the Master/Slave-mode in order to create a new connection. This makes the devices discoverable with their MAC-address free for anyone listening in on the wireless unencrypted traffic.

Chapter 7

Discussion

Privacy is an important factor to any protocol as it is closely linked to security, but not necessarily. There are implementations that can have great privacy protection, but not good security and of course the opposite. Even with great security, any protocol can leak information that ultimately weakens the privacy of the protocol.

General Data Protection Regulation (GDPR) proved that digital data needs to be protected to improve privacy, which should imply wireless data, but is considered hard to enact and legislate. The same comes to legislating privacy laws, as it is illegal to store MAC-addresses gathered from both clandestine and active location tracking.

The idea behind this thesis is that if you manage to successfully gather a unique identifier of UE, the MAC-address is one such identifier, then exists the possibility of tracking that individual or the carrier of that UE.

As the wireless protocol is described in Chapter 2 IEEE 802.11 Wireless, we can understand that the wireless protocol is complicated and contains multiple fields in the headers. In earlier version of Android and iOS, the probe used to contain the SSID of the top connected routers of the UE. This made it possible to find common places the UE visits, as the possibility of using open source logging engines such as the site *wigle* exists, see Section 2.3 Wi-Fi Probe Sniffing. In reality this would mean a probe sniffer could find out the home and workplace of the owner of the UE.

In later version of Android and iOS the SSID was no longer included in the probe request together with randomization of the MAC-address, which made this type of clandestine tracking no longer effective. The randomization is only implemented when sending probe requests, and not during the connection phase. Thus revealing the real address of the UE. In Android 9.0 a developer option has been implemented randomize the MAC-address even during the connection phase. The reason why this should be implemented in the standard and not just as a developer option is because of privacy. The negative side of this is that in WLANs each UE has to have a unique address to be able to receive the correct packages from routing. On the other hand the possibility of two addresses to be equal are 1 to $16^{1}2$ as there are twelve hexadecimals, which is 1 to 281'474'976'710'656. That is approximately 1 in 281 trillion chance, which in reality is next to impossible of it occurring.

The bluetooth protocol is described in Chapter 3 Bluetooth, which includes only relevant and small parts of the total capability and options of the protocol. The bluetooth protocol is much more advanced and sophisticated compared to the IEEE 802.11 standard. Thus making it harder to create a successful attack to compromise the protocol, which is part of the limitations.

From the results found in Chapter 5 Results, we have proved that clandestine and active location tracking is very simple using open source software and cheap hardware. The clandestine location tracking from Section 5.1 Passive Location Tracking, had the goal of gathering MAC-addresses from the UE, and very much succeeded in doing so. The prerequisites required a good literature study before conducting the experiments, which in the end gave great results.

In Section 5.2 Active Location Tracking the results from active Wi-Fi sniffing proved that affordable equipment, such as a Hak5's Pineapple Nano (See Subsection 4.1.3 Hak5 WiFi Pineapple Setup), can obtain the currently connected AP of the UE. With my understanding of the Wi-Fi protocol the connections in WPA2 are encrypted, hence you would have to break the PSK to decrypt any traffic, which proved to be true to some degree. What proved to be wrong on the other hand was that the source and destination address were encrypted. The wireless packages apparently include the source and destination address unencrypted in the data link header, which is the second layer in the OSI BRM-model. This made this objective easier than expected.

For eavesdropping the results from MiTM attacks against the WPA2 protocol, as described in Section 5.3.1 Wireless MiTM Attack, proves that with weak PSK the routers can be hacked. Hacked routers, or cracked PSKs in this case, opens the possibility for MiTM. In reality this opens the door for eavesdropping on the traffic between the UE and the AP. Even though most sites have implemented HTTPS, it is still not enforced on all sites. Downgrade attacks can then attempt to force the use of HTTP, making eavesdropping possible again. It is important to note that this method of attack has been mitigated in WPA3, all attempts at guessing the PSK must now go directly with the AP. This is because of the implementation of SAE.

For eavesdropping on bluetooth, we did not manage to get any results except from sniffing L2CAP-packages encrypted with a LTK. During the literature study, multiple options for bluetooth MiTM was found. Only one of these were carried out, but not giving expected results as described in the study. This can be explained by the prerequisites set in the first phase of this thesis. The technical and theoretical understanding was most probably lacking for this attack to be successful, as it is highly sophisticated and very difficult to successfully exploit the bluetooth protocol. The software version of the protocol could explain the difficulties experienced, as bluetooth receives regularly updates towards its security.

The use of certificates and public/private-key encryption could greatly improve the security, privacy and authentication of devices. The downsides of implementing certificates are the size of the handshake and calculation, in some cases when using Internet-Of-Things (IOT) devices or BLE with limited battery consumption, which could limit the battery lifetime. Thus comes the trade-off between security and privacy versus usability and availability.

There are multiple ethical questions that can be discussed in this thesis. One of the most important ethical questions in this thesis is the act of gathering MAC-addresses. This is illegal if not done correctly and made anonymous by implementing hashes or similar measures. Thus any information that could be used to identify an individual is made anonymous in this thesis by blurring images and deleting logs. The author of this thesis decided not to include a step-by-step guide for the 5.3.1 Wireless MiTM Attack, as this is not the focus of the thesis, but rather the problems that persists from this type of attack.

Chapter 8

Conclusion

The thesis has covered some sections of the protocols bluetooth and Wi-Fi with regards to privacy. The discussions include arguments weighting both the positive and the negative impacts for the decisions made during experiments, which are backed by research. The results from the experiments shows that both clandestine and active location tracking of individuals is fairly easy, without the need of a very technical understanding. Eavesdropping requires advanced and highly sophisticated attacks, making this harder to carry out even though it could be possible.

As the discussions have covered, there are far too many possibilities of both clandestine and active location tracking, to even consider the protocols as privacy protected. The suggested countermeasures provides both simple and advanced solutions as they require from little redesign to a complete redesign of the both the connection handshake and encryption respectively.

The recommended solutions would be to implement total randomization of the MAC-address for each new connection and the use of public-key cryptography. Both solutions would prevent clandestine and active location tracking, but only public-key cryptography would mitigate MiTM-attacks against both protocols.

With public-key cryptography the bluetooth connection handshake, see Figure 3.1, would be able to encrypt data from the very first package sent. Thus making us able to go directly to phase 4, with the ability to confirm the identity of the device by simply checking the signature of the public certificate. Downside of implementing certificates is when using IOT devices or BLE, which would limit the speed and lifetime because of the needed package size, but would increase the security and privacy. Thus having the trade-off between security and privacy versus usability and availability in this case.

From the discussion we can conclude that even with good research and prerequisites, the experiments can in some cases be difficult and easier on paper than in reality. This made the experiments very time consuming and suggests for better planning for future work when conducting MiTM-attacks against bluetooth.

In total the thesis managed to reach almost all four objectives, set in the first phase of this thesis, with only the Wi-Fi part successful in objective three.

8.1 Future work

Future work could include an experiment with IMSI-catching using Wi-Fi as described in Section 2.3 Wi-Fi Probe Sniffing. In Section 5.3.2 Bluetooth MiTM Attack, an attack towards bluetooth in regards to attempting MiTM was not successful, and would need further research in future work. Any attempt at creating new software or hardware as well as using existing ones is recommended towards active and clandestine attacks on WPA3. The protocol was recently released, as of writing this thesis, and would be interesting to see tested in future work. Lastly, both a theoretical and practical approach to the implementation of certificates, in an effort to enhance privacy, is recommended for future work.

Bibliography

[1] Harshita Pandey Kundana Thiyari. Computer network | layers of osi model. https://www.geeksforgeeks.org/layers-osi-model/, 2018. Accessed: 2019-02-05.

[2] Datatilsynet. Tracking in public spaces. the use of wifi, bluetooth, beacons and intelligent video analytics. https://www.datatilsynet.no/globalassets/global/om-personvern/rapporter/sporing-i-det-offentlige-rom_eng_web.pdf, 2016. Accessed: 2019-02-05.

[3] macaddresschanger. What is bluetooth address (bd_addr). https://macaddresschanger.com/what-is-bluetooth-address-BD_ADDR, 2017. Accessed: 2019-05-13.

[4] bettercap. Bettercap image of ui-events. https://raw.githubusercontent.com/bettercap/media/master/ui-events.png, 2019. Accessed: 2019-02-08.

[5] nightwatchcyber. Sensitive data exposure via wifi broadcasts in android os [cve-2018-9489]. https://wwws.nightwatchcybersecurity.com/2018/08/29/sensitive-data-exposure-via-wifi-broadcasts-in-android-os-cve-2018-9489/, 2018. Accessed: 2019-04-03.

[6] Mohit Kumar. German police seek help in finding parcel bomber with mac address. https://thehackernews.com/2019/01/german-dhl-parcel-bomb-blackmailer.html, 2019. Accessed: 2019-01-20.

[7] Mohammed Alloulah, Anton Isopoussu, Chulhong Min, and Fahim Kawsar. On tracking the physicality of wi-fi: A subspace approach. *IEEE Access*, 7:19965–19978, 2019.

[8] Sean Carlin and Kevin Curran. An active low cost mesh networking indoor tracking system. *International Journal of Ambient Computing and Intelligence (IJACI)*, 6(1):45–79, 2014.

[9] Eric M Hutchins, Michael J Cloppert, and Rohan M Amin. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. *Leading Issues in Information Warfare & Security Research*, 1(1):80, 2011.

[10] Android. Privacy: Mac randomization. `https://source.android.com/devices/tech/connect/wifi-mac-randomization`, 2017. Accessed: 2019-05-15.

[11] Apple. Network security, 2018.

[12] SINTEF. Are hellandsvik. `https://www.sintef.no/en/all-employees/employee/?empId=4501`, 2019. Accessed: 2019-05-27.

[13] SINTEF. Christian frøystad. `https://www.sintef.no/en/all-employees/employee/?empId=5350`, 2019. Accessed: 2019-05-27.

[14] Julien Freudiger. How talkative is your mobile device?: an experimental study of wi-fi probe requests. In *Proceedings of the 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, page 8. ACM, 2015.

[15] Brian P Crow, Indra Widjaja, Jeong Geun Kim, and Prescott T Sakai. Ieee 802.11 wireless local area networks. *IEEE Communications magazine*, 35(9):116–126, 1997.

[16] Wi−Fi.org. Wi-fi alliance® introduces security enchancements. `https://www.wi-fi.org/news-events/newsroom/wi-fi-alliance-introduces-security-enhancements`, 2018. Accessed: 2019-05-16.

[17] Wi-Fi Alliance. Discover wi-fi security. `https://www.wi-fi.org/discover-wi-fi/security`, 2018. Accessed: 2019-06-17.

[18] Enisa. Passive wifi surveillance and access point hijacking. `https://www.enisa.europa.eu/publications/info-notes/passive-wifi-surveillance-and-access-point-hijacking`, 2015. Accessed: 2019-02-16.

[19] wigle. Wireless geographic logging engine. `https://wigle.net/`, 2019. Accessed: 2019-05-31.

[20] Tom Spring. Wi-fi hotspot finder spills 2 million passwords. `https://threatpost.com/leaky_app_data/144029/`, 2019. Accessed: 2019-05-31.

[21] The Hacker News. Wi-fi can be turned into imsi catcher to track cell phone users everywhere. `https://thehackernews.com/2016/11/imsi-track-cellphone.html`, 2016. Accessed: 2019-02-16.

[22] Bluetooth SIG Proprietary. Specification of the bluetooth system, v5.1. `https://www.bluetooth.org/docman/handlers/downloaddoc.ashx?doc_id=457080`, 2019. Accessed: 2019-05-13.

[23] Mark Loveless. Understanding bluetooth security. https://duo.com/decipher/understanding-bluetooth-security, 2018. Accessed: 2019-03-20.

[24] putridparrot. The bluetooth device address (bd_addr). http://putridparrot.com/blog/the-bluetooth-device-address-bd_addr/, 2018. Accessed: 2019-05-13.

[25] raspberrypi.org. Raspberry pi. https://www.raspberrypi.org/, 2019. Accessed: 2019-06-18.

[26] kismetwireless. Kismet. https://www.kismetwireless.net/, 2019. Accessed: 2019-05-27.

[27] greatscottgadgets. Ubertooth one. https://github.com/greatscottgadgets/ubertooth/wiki/Ubertooth-One, 2017. Accessed: 2019-02-05.

[28] greatscottgadgets. 2018-12-r1 - the uberducky release. https://github.com/greatscottgadgets/ubertooth/releases, 2018. Accessed: 2019-03-15.

[29] bettercap. The swiss army knife for 802.11, ble and ethernet networks reconnaissance and mitm attacks. https://github.com/bettercap/bettercap, 2019. Accessed: 2019-02-08.

[30] bettercap. Bettercap modules. https://www.bettercap.org/modules/, 2019. Accessed: 2019-05-27.

[31] bettercap. The swiss army knife for 802.11, ble and ethernet networks reconnaissance and mitm attacks. https://github.com/bettercap/bettercap/issues/468, 2019. Accessed: 2019-02-08.

[32] Gattack. Gattack outsmart the things. https://gattack.io/, 2019. Accessed: 2019-02-08.

[33] mikeryan. Crack and decrypt ble encryption. https://github.com/mikeryan/crackle, 2019. Accessed: 2019-02-08.

[34] darknet. crackle – crack bluetooth smart encryption (ble). https://www.darknet.org.uk/2017/02/crackle-crack-bluetooth-smart-encryption-ble/, 2017. Accessed: 2019-02-08.

[35] Adafruit. Bluefruit le sniffer - bluetooth low energy (ble 4.0) - nrf51822 - firmware version 2. https://www.adafruit.com/product/2269, 2018. Accessed: 2019-02-08.

[36] Nordic Semiconductor. nrf sniffer. https://www.nordicsemi.com/Software-and-Tools/Development-Tools/nRF-Sniffer, 2019. Accessed: 2019-02-08.

[37] GNU Radio. About gnu radio. https://www.gnuradio.org/about/, 2019. Accessed: 2019-02-08.

[38] Digital Security. Btlejuice bluetooth smart (le) man-in-the-middle framework. https://github.com/DigitalSecurity/btlejuice, 2018. Accessed: 2019-02-08.

[39] virtualabs. Bluetooth low energy swiss-army knife. https://github.com/virtualabs/btlejack, 2019. Accessed: 2019-02-08.

[40] Alchemistowl. Pocorgtfo17. https://www.alchemistowl.org/pocorgtfo/pocorgtfo17.pdf, 2017. Accessed: 2019-04-25.

[41] Hak5. Wi-fi pineapple. https://www.wifipineapple.com/, 2019. Accessed: 2019-04-03.

[42] Electronic Notes. Bluetooth radio interface, modulation, channels. https://www.electronics-notes.com/articles/connectivity/bluetooth/radio-interface-modulation-channels.php, 2019. Accessed: 2019-05-13.

[43] Mathy Vanhoef, Célestin Matte, Mathieu Cunche, Leonardo S Cardoso, and Frank Piessens. Why mac address randomization is not enough: An analysis of wi-fi network discovery mechanisms. In *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*, pages 413–424. ACM, 2016.

[44] Jeremy Martin, Travis Mayberry, Collin Donahue, Lucas Foppe, Lamont Brown, Chadwick Riggins, Erik C Rye, and Dane Brown. A study of mac address randomization in mobile devices and when it fails. *Proceedings on Privacy Enhancing Technologies*, 2017(4):365–383, 2017.