Detection, Mapping, and Tracking of IoT Devices Using Bluetooth

Ruhit Rafian Prinon

Thesis submitted for the degree of
Master in Informatics: Information Security
60 credits

Department of Informatics
Faculty of Mathematics and Natural Sciences

UNIVERSITY OF OSLO

Autumn 2023

Detection, Mapping, and Tracking of IoT Devices Using Bluetooth

Ruhit Rafian Prinon

Detection, Mapping, and Tracking of IoT Devices Using Bluetooth

http://www.duo.uio.no/

Printed: Reprosentralen, University of Oslo

Abstract

This thesis aims to detect, track, and map IoT devices in an open-world environment, identify potential privacy and security risks that these devices are vulnerable to, and develop ways to address them. The purpose here is to document an understanding of what data is being transmitted by these IoT devices, how the data can be interpreted, and what inferences can be made from it. It focuses on on-person IoT devices that use Bluetooth, including wireless earbuds, fitness trackers, smartwatches, and smart glasses. This study relies on Bluetooth triangulation and primarily uses the Received Signal Strength Indicator (RSSI) variable. RSSI is used to passively track a device in an open-world environment using 3 or more Bluetooth scanning devices. The hypothesis is to prove that it is feasible to precisely track a victim in an open-world environment using RSSI readings emitted by the IoT devices worn by the victim. Furthermore, this study relies on the use of location estimation algorithms such as trilateration and triangulation and implements a learning algorithm to improve accuracy.

Further future work in the field of on-person IoT device tracking is also discussed. Additionally, Bluetooth security and privacy are addressed. The vulnerabilities of Bluetooth devices are discussed in details and solutions to these issues are suggested.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to express my sincerest gratitude to both my supervisors, Professor Nils Gruschka and Professor Lothar Fritsch from the University of Oslo for their time and efforts in making this thesis possible. Their insights into the topics of digital security and privacy as well as in-depth knowledge of technology aided in a more effective study. They are true pioneers of the industry and their attention to detail when it comes to research and experimentation has guided me throughout the entire study process. I value their feedback and their advice and I am grateful to have had such great mentors throughout my thesis.

I would also like to thank my friend who was willing to be the anonymous participant to act as a subject during the experimentation.

On a personal note, I would like to thank my parents, who, even though are thousands of miles away, are always with me and next to me. Last but not least, I wish to convey my special thanks to Pooja Aryal and Sefat Noor Orni for their moral support and for making Oslo feel like home.

List of Abbreviations

IoT	Internet of Things
UHF	Ultra High Frequency
WiFi	Wireless LAN
GPS	Global Positioning System
API	Application Programming Interface
RSSI	Received Signal Strength Indicator
dBm	Decibel-Milliwatts
RF	Radio Frequency
RF	Radio Frequency
BLE	Bluetooth Low-Energy
AoA	Angle of Attack
AoD	Angle of Departure
SSID	Service Set Identifier
MAC	Media Access Control
ISP	Internet Service Provider
RAM	Random Access Memory
JS	JavaScript
CPU	Central Processing Unit
USB	Universal Serial Bus
CLI	Command Line Interface
CSV	Comma Separated Values
HID	Human Interface Device
JS	JavaScript
UI	User Interface
SSH	Secure Shell
FTP	File Transfer Protocol
GDPR	General Data Protection Regulation
CCPA	California Consumer Privacy Act
URL	Uniform Resource Locator

Chapter 1

Introduction

This chapter contains a short introduction to the thesis, its goals, research questions, and structure.

1.1 Motivation

IoT devices are ever-present in a modern urban environment. This includes on-person devices such as smart watches, wireless earphones, and fitness trackers. It also includes a collection of smart home equipment that provides security, accessibility, and comfort to the user. These devices, by their very nature, have to be wireless to provide the promised level of convenience. However, in doing so, the somewhat *open* nature of Bluetooth provides malicious actors with another possible attack vector. While almost all IoT devices use some form of wireless communication on the 2.4GHz UHF radio wavelength, on-person IoT devices use Bluetooth due to being more energy-efficient. On the other hand, in-home smart equipment usually goes with WiFi. Since only Bluetooth is used in on-person IoT devices, only the Bluetooth protocol will be explored for the purpose of this thesis.

On-person Bluetooth devices include wireless earbuds, fitness trackers, smartwatches, and novel smart glasses as well. Wireless earbuds are used for media consumption, voice assistants, hands-free voice calling, and so on, while fitness trackers can be used for tracking various physical activities, measuring biological data like heart rates, blood oxygen saturation (SpO_2), and so on. Smartwatches usually have a larger list of functionalities such as telecommunication, contactless payment, and GPS navigation along with all the features of a fitness tracker.

1.2 Bluetooth Security and Privacy

Bluetooth security and privacy are critical considerations when it comes to IoT devices. One of the primary security risks associated with Bluetooth is device spoofing. Attackers can use spoofed devices to mimic legitimate Bluetooth devices, allowing them to intercept and manipulate data transmitted between devices. Attackers can also use Bluetooth vulnerabilities to gain unauthorized access to IoT devices or launch denial-of-service attacks.

To mitigate these risks, Bluetooth technology includes various security features, such as authentication and encryption protocols as well as MAC randomisation to protect the confidentiality and integrity of data transmitted between devices. However, due to poor implementation of the Bluetooth protocol by vendors or developers, the security measures can be compromised.

Additionally, Bluetooth technology can pose significant privacy risks as it broadcasts device information such as device name, address, and capabilities. This information can be used to track the location of the device and the user, providing attackers with sensitive information that can be used for malicious purposes.

1.3 Background in Bluetooth

Bluetooth devices are configured by the Bluetooth protocol to advertise their existence during operation. According to the book "*Bluetooth Essentials for Programmers*" by Huang and Rudolph [17], each of these advertisements contains information required to create a connection as well as to uniquely identify the device. Similar to how internet protocol devices are physically identified, MAC addresses are also used in Bluetooth to uniquely identify a device. Each vendor or manufacturer is assigned a set of MAC addresses by the IEEE authorities that they can uniquely assign to a device. Additionally, every time the advertisement packet is received, the recipient device calculates the RSSI (Received Signal Strength Indicator) value based on the strength of the RF wave received.

Received Signal Strength Indicator is one of the core components of Bluetooth triangulation, which is measured in decibel-milliwatts (dBm). This variable defines how strong the quality of the radio signal is between 2 devices. Therefore, due to the natural path loss or path attenuation of electromagnetic waves, the RSSI value and the proximity between the devices are correlated. Consequently, it is feasible to create a path loss model for the two variables RSSI and Distance. However, according to authors at Bluetooth.com [16], the RSSI values are relativistic and different devices have different ranges of values that will be returned from the high-level API. It also mentions that even at the same distances, different devices will return different dBm values due to discrepancies in Bluetooth chips and RF circuit designs. It is therefore apparent that a learning process along with a learning algorithm is first needed to calibrate the system for inaccuracies and fluctuations due to the environment.

1.4 Initial Hypothesis

With the background of Bluetooth in mind, this study will attempt to leverage this RSSI-Distance relationship and test whether it is feasible to locate a user in an open-world environment. The purpose here is to prove that external actors can use some form of Bluetooth scanner to locate devices such as smart watches, wireless earbuds, smart glasses, or even Bluetooth media players in a car. Additional information gathered and

discussed in the background research chapter will allow us to build a more detailed hypothesis.

1.5 Structure

In this thesis, Chapter 2 includes a study into the state of the art of Bluetooth location sensing, data mining as well as privacy and security analysis of IoT devices. We then use the understanding to build a more detailed hypothesis, plan a project and describe its dependencies as well as the equipment required. Chapter 4 describes the use of experimentation to understand the equipment as well as the software. The results of these pre-experiments are then used to design a final experiment that tests the hypothesis proposed in Chapter 3. Chapter 5 dives into the analysis of the data from the final experiments; it explains the implementation of data pre-processing, the development of a distance estimation model that describes the relationship between RSSI and distance, and finally, an efficient trilateration algorithm that can be used to estimate victim position. Chapter 6 discusses the privacy implications of this work, threat protection as well as limitations and ethical considerations. Finally, Chapter 7 concludes with a summary of the results and possible future opportunities in this topic.

Chapter 2

Background

In this chapter, the state of the art in Bluetooth and IoT device security is discussed. Along with that, the discussion also includes references to modern techniques in signal triangulation and trilateration as well as the extraction of identity attributes from devices in the wild.

2.1 Previous Work in Triangulation

There have been several different attempts in the past to build a Bluetooth-based locator system, and most of them usually focus on indoor location determination. In this section, we will look into how these studies took place, what devices and processes were used and how feasible the locating techniques are.

The use of a Received Signal Strength Indicator to triangulate a device's relative position has been demonstrated by Wang et al. [29] where they use 3 Bluetooth devices to determine the relative position of a target device. Their research explored the indoor positioning of Bluetooth devices where all the devices were fitted with standard Bluetooth 2.1 transceivers. Their approach relies on measuring the RSSI readings using 3 Android-powered devices (henceforth referred to as **base stations**) inside a room. The RSSI values are read from the base stations and then processed using an RSSI propagation model that ideally would produce the exact coordinates for the target Bluetooth device. But due to inaccuracies and propagation errors, distances are found to be in a range of values instead. This range of values can be then fed into 3 different standardized techniques for distance triangulation:

1. Least Square Estimation: Perhaps one of the most widely used wireless positioning techniques [9], it relies on eliminating propagation and detection errors in the distance values we get from the RSSI propagation model in order to create a line of best fit. This can then be used to estimate a more accurate distance reading that is more precise than the initial values.

2. Three Border Positioning: Since the RSSI propagation model gives us 3 different values for distance from the 3 base stations, a set of Euclidean distance equations can be solved to find the average location of the mobile device.

3. Centroid Positioning: The centroid positioning technique is slightly different from the rest with regard to the number of base stations used. Instead of 3, it uses 4 different base stations. The 4 intersection points between each adjacent base station are then used to deduce the precise location of the target device.

In the end, the authors conclude by showing that Least Square Estimation provides slightly better results in terms of location estimation, but they also mention that if accurate enough RSSI readings are recorded, any of the 3 will yield good results.

Wang et al. have also explored the effects of signal strength on external conditions such as the human body. According to their experiments, have found that there is a clear reduction of signal strength by 6-8 dB when the human body is covering parts of the antenna.

Following the work by [29], Li et al. [21] propose real-time corrections to RSSI values to further improve accuracy. The authors hypothesize that Bluetooth transmission strength, especially in high-efficiency mobile devices, constantly fluctuates to create ever-changing RSSI values for even stationary objects. Moreover, since RSSI values are inaccurate due to fluctuating transmission power of the Bluetooth antennas, the distance values are also inaccurate and tend to change. Therefore, in the first part of the paper, the authors analyse the fluctuations of Bluetooth base stations in order to find patterns in said fluctuations. They find that devices provide variances of up to 10 dBm even while standing still. Moreover, they also argue that simply increasing the number of base stations adds to the processing time required and adds latency to an otherwise time-efficient process. They also argued against the use of Kalman filters that have been explored by other researchers on the same premise. Therefore, they proposed a real-time correction algorithm using a back propagation neural network (BPNN) and perform a series of experiments to reduce the inaccuracies caused by power fluctuations.

Finally, their proposed model includes the use of a PSO-BPNN model to improve the distance data collected which is then fed into a Least Square Algorithm [9] to provide the location data. They end the paper by doing a comparative study of the different existing methods as well as their proposed model and show that the accuracy of their model significantly surpasses that of the previous work.

Another work by Bai et al. [3] proposes a BLE-based indoor positioning system that can track users in their home environment in order to be used for health analysis. In their work, several Bluetooth base stations, are spread across a home such that, at any point, a user will be in range of at least 3 Bluetooth stations. The authors use several Raspberry Pis with Bluetooth 4.0 LE modules as the base stations and a wearable BLE base station that the user would wear in this scenario. They use 2 different methods to do the position calculation. However, only the first method is relevant to the work in this thesis, and therefore the second method is not explored in further detail:

- Step 1: RSSI values are read from the base stations at specific intervals and then fed into a Kalman filter in order to smooth out the discrepancies and variances. This outputs RSSI values to the next step.

- Step 2: The distance values are then fed into a Path Loss Model which equates the relationship between RSSI and distance. This step outputs 3 distance values to the next step.
- Step 3: The distance values are taken and then fed into a Trilateration-based algorithm where 3 Euclidean equations much like the Three Border Positioning in [29] solved to find the exact Cartesian coordinates of the Bluetooth base station, and hence the user.

Faragher et al. [14] proposed the use of Bluetooth Low Energy scanners to deduce the location of Bluetooth devices in an indoor environment. They perform experiments using several Bluetooth scanners and varying several attributes in the process such as scanner distribution, scan rate, and using Wi-Fi instead of Bluetooth. Their best results included an accuracy of less than 2.6m through 95% of the tracking time. They have deduced very interesting findings that will be useful for this project such as:

1. Scan rates above 10Hz provide diminishing returns
2. There is an increase in accuracy to up to 8 scanners in the same area, but it increases processing time greatly
3. Scanner transmission power of -15dBm provided the best coverage in an indoor environment

2.2 Previous Work in Location Sensing Techniques

Research has also been done on the use of directional antennas to precisely locate Bluetooth devices in indoor environments. This technique involves highly sensitive directional antenna arrays that can focus their scans in a specific direction. The directional antennas can provide a more focused signal and eliminate interference from surrounding devices, improving the accuracy of location detection. For instance, a study by Jia et al. [20] demonstrated that Bluetooth location tracking using directional antennas can achieve an accuracy of 1.42 meters in indoor environments.

The newest Bluetooth Specification, 5.3 [12], describes the addition of new protocols that can sense the location of a Bluetooth device by using several antennas in an indoor environment. Angle of Arrival (AoA) and Angle of Departure (AoD) are methods used in Bluetooth direction finding to determine the location of Bluetooth devices in three-dimensional space.

AoA is a method where the receiver measures the angle between the device and the antenna array and uses this information to calculate the device's location. This method requires that the device has a highly directional antenna that can be pointed directly at the receiver.

AoD, on the other hand, is a method where the device measures the angle between its own antenna and the receiver's antenna array and uses this information to calculate the receiver's location. This method requires that the receiver has a highly directional antenna that can be pointed directly at the device.

Both methods require that the antennas used have a high level of accuracy and precision and that they can handle the high-frequency signals used in Bluetooth communication. These methods are used to enable precise location-based services, such as asset tracking and indoor navigation in large complexes such as airports, train terminals, and supermarkets.

2.3 Relevant Work in Tracking and Data Mining

Fritsch et al. [15] have performed a similar experiment to prove that it is possible to extract and accumulate identity attributes from IoT devices in an ambient environment. Experiments using a wireless signal (Bluetooth and Wi-Fi) scanning app called "WIGLE" was performed to detect and accumulate IoT devices in 4 target locations throughout Oslo, Norway. The 4 locations proved to have different degrees of success due to the density of the population (and thus, smart devices), the fleeting nature of pedestrian devices and cars passing by, and so on. However, it was found that it is relatively easy to find and make inferences from IoT devices whose SSID have not been anonymised. This SSID can be used to make inferences such as:

1. whether the owner of the house is present (if a smartwatch or fitness bands are present)
2. the socioeconomic status of the household or neighbourhood (depending on the price of the devices detected)
3. name of the owner (Apple devices add the first name of the owner onto their device's SSID by default)

In a 2019 work by Becker, Li, and Starobinski [4], it was shown that continuous tracking of Bluetooth 4.0-compliant devices is still possible with the use of identifying tokens. Even though with the advent of MAC

randomization, it is difficult to continuously and trivially track a device through the MAC address, it was shown that with the help of secondary identifiers along with a novel *address-carryover algorithm*, it is possible to track smart devices. The authors also have managed to find other privacy vulnerabilities that exist when iOS devices communicate with connected IoT devices.

In this article [28] by Versichele et al., they have explored a novel approach to detect movement amongst tourists in a specific city and have tried to create a pattern map of what tourist attractions and in what order the tourists choose to visit in a given day. For the city, Ghent, Belgium was chosen and 29 different tourist attractions were chosen. In each location, Bluetooth scanners were placed for 15 days, recording the MAC address and COD (Class of Device) code was recorded along with a detection timestamp. The results were then filtered through to eliminate non-tourist detection, followed by using association rule learning to create a visit pattern map. The authors show how Bluetooth fingerprinting can be used by cities or tourism companies to find valuable personalized tracking data, but on the other hand, poses a risk to privacy by creating identity attributes of even passive devices carried by tourists.

In another work exploring the privacy concerns of Bluetooth, Valeros et al. [27] have used the popular technique called "Bluedriving" to answer privacy-related questions. They used this technique to detect approximately 3000 devices throughout 5 cities in South America and Europe. Through their work, they were able to make several privacy-infringing inferences from their captured devices such as finding medical devices, movement patterns over several days, whether a person is home and so on, highlighting the ease of use of this attack vector.

2.4 Security and Privacy Analysis

Several studies have been conducted on the vulnerabilities of Bluetooth technology and the potential risks associated with the use of Bluetooth-enabled IoT devices. For example, a study conducted by Antonakakis et al. [1] investigated the security of Bluetooth Low Energy (BLE) devices and found that many devices were vulnerable to attack due to poor implementation of the BLE protocol. Another study by Kim et al. [19] demonstrated how Bluetooth-enabled devices could be exploited to launch large-scale attacks against wireless networks.

This paper by Loi et al. [23] explores the security vulnerabilities of several consumer IoT devices and evaluates the overall strength of the

defences put up by the IoT devices against active attacker situations. The security tests contain standard attacks against the Confidentiality, Integrity, and Availability of a device along with its capacity to deter reflective attacks. The authors concluded from their testing that all the devices have one or more forms of a security vulnerability with varying levels of success. They propose that as IoT devices become ever-present in our lives, the use of a rating system for defensive security capabilities (much like energy efficiency ratings) would offer consumers more control over what devices they trust their data with.

Similarly, Notra et al. [24] have studied the network activities of 3 household devices using Wireshark to find security vulnerabilities. During their experimentation, they found several attacks that can infringe upon the user's security and privacy. To that end, they also provide solutions to all of the issues that rose during their research, indicating that all the security vulnerabilities have relatively low-effort solutions that the manufacturers, ISPs, or even users can build into smart devices if they so choose.

Numerous studies have explored the use of trilateration algorithms for positioning in IoT applications. For instance, researchers have proposed using trilateration with Bluetooth Low Energy (BLE) technology to determine the location of an object in a warehouse environment with an accuracy of about 30 cm [22].

Chapter 3

Project Description

This chapter explores the possibilities of experimenting with Bluetooth signals and attribute capture using specialised hardware. It also outlines requirements and a hypothesis of the experiment

3.1 Detailed Hypothesis

Following the research into the state of the art, we can now formulate a hypothesis and create a plan to test and prove the hypothesis in a systematic manner. Bluetooth Low Energy (BLE) [7] is a popular wireless communication technology for IoT devices due to its low power consumption and simplicity of implementation. BLE devices transmit packets of data that include RSSI values, which can be used to estimate the distance between two devices if we capture the data using scanners. These devices are generally efficient and sensitive enough for an attempt at triangulation. Therefore, we can conclude by stating some hypotheses that we will test out with our experimentation and data analysis:

- Hypothesis 1: It is possible to estimate a victim's location in an open-world environment using their Bluetooth broadcasts
- Hypothesis 2: The sphere of detection of these victims is rather large and can be extended further with sensible upgrades
- Hypothesis 3: Building such a scanning setup does not involve any custom parts or rather expensive equipment
- Hypothesis 4: The location is accurate enough to viably spy on victims and gain insights into movement, activities and routines

3.2 Procedure and Processes Involved

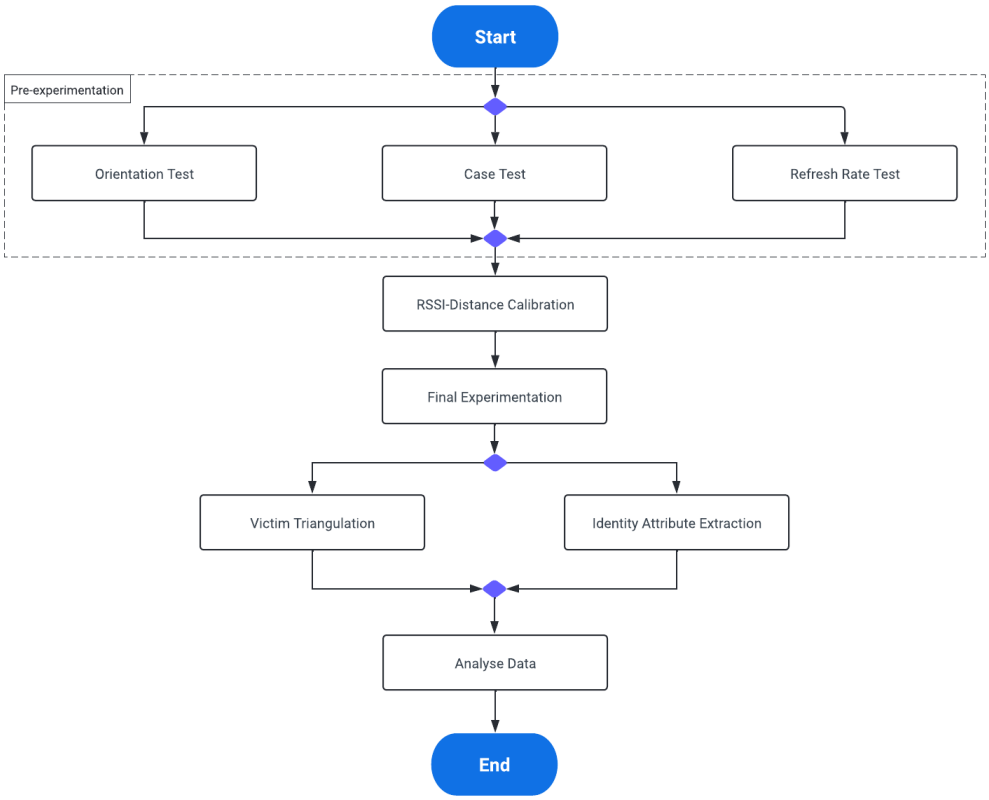

Figure 3.1: Flowchart showing the sequence of stages and processes of this project

This flowchart represents the order and design of the overall process we will use to prove our hypothesis. At first, we will run some pre-experiments to gather more information about the equipment as well as the Bluetooth protocol. These pre-experiments will be used to determine the most optimal way of using the hardware and software associated with the context of the project. Following these experiments, we can then run calibration experiments in order to understand the relationship between RSSI and distance. The data collected during calibration runs are then used to build a learning algorithm that can determine the distance of a device based on the RSSI value received from it. Following that, we can finally move on to the final experiment where we collect RSSI values of all devices in an outdoor environment from 3 Bluetooth scanners. Finally, we can use the data collected to attempt victim location triangulation as well as identity attribute extraction. These results will then be analysed to determine whether and to what extent the security and privacy of the Bluetooth protocol can be abused.

3.3 Experiment Design

Generally, mainstream operating systems limit Bluetooth scanning intervals as well as the number of detected devices. This is done for safety purposes and as a power-saving measure. It is evident that the experiment into triangulation will require special hardware upon which limitations have not been imposed. For the experiment, 3 Bluetooth scanners will be used to scan for devices. Along with that, other supporting hardware will be used to ensure that the Bluetooth scanners (henceforth referred to as **base stations**) work as intended.

To perform this experiment, 3 Base Stations are placed in an open space with a relatively large number of possible target devices. The base stations will constantly scan for Bluetooth devices at a certain refresh rate and record several variables required for the triangulation calculation and for possibly generating identity attributes. The list of variables that will be recorded is as follows:

1. RSSI
2. MAC Address
3. Vendor
4. Detected at
5. Alias

As we are using RSSI to estimate distance, it is worthwhile to know the relationship between RSSI and distance. Therefore, a series of pre-experiments must be performed in a controlled environment in order to create a better understanding of the correlation between RSSI and distance. This pre-experiment along with others will be discussed further in Section 4.1.

3.4 Base Station Devices

During operation, each device in range will generate a record at each of the base stations at each scan refresh. If there is a large number of devices in range, the data set may end up being quite large. As the data is multi-faceted and has several dimensions, there is potential for the files to be quite large and will require more storage as well as RAM to be processed. Therefore, the base stations that we are using should be able to handle such high data throughput and store large amounts of data in the persistent memory for later offloads and analysis. Therefore, 3 such devices were

proposed and their specifications were analysed to find which is the most appropriate for our use case:

Arduino Due

Advantages: Power efficient, Easy to program, Versatile
Disadvantages: Limited CPU and Memory, Lack of built-in RF Radio

Arduinos are an open-source line of microcontrollers with a customised ATmega328P microcontroller as the CPU. It is a very versatile tool that can be programmed using the Arduino IDE. For our case, the most relevant version is the Arduino Due[2], the most capable device both in terms of I/O and the CPU. It is extremely power efficient with a maximum board power consumption of 800mA on 3.3V while the I/O pins can consume another 130mA. The microcontrollers are also quite simple to program; due to it being an entirely open-source project, there is a large library of resources from other researchers and hobbyists. However, the biggest limitation of the device is the meagre onboard memory capacity and CPU throughput. Along with that, the lack of a built-in RF Radio means that an external Bluetooth module is required to turn the device into a Bluetooth scanner. Moreover, since it is not internal, extra shielding may be required to shield the antenna from the exposed circuits of the Arduino.

Raspberry Pi

Advantages: Easy to program, Capable CPU and Memory, Built-in BLE Module
Disadvantages: Limited Bluetooth sensitivity, Moderate power requirements

Raspberry Pis are popular single-board computers with full-sized desktop OS support. For our application, we analysed the Raspberry Pi Model 3B+ [25] which has a very capable 1.4GHz 64-bit quad-core processor. They can be very versatile for a large range of use cases due to the desktop OS support. The Raspberry Pi also comes with an onboard WiFi and Bluetooth antenna, and therefore it is not required to use external antennas in order to get Bluetooth scanning features. Due to its higher-end CPU, it also requires more power and thus a capable power source has to be determined for portability.

ESP32-S3

Advantages: Power efficient, Built-in Bluetooth module, Capable CPU
Disadvantages: Limited Memory capacity, Complexity, Lack of community support

The ESP32-S3 [13] is a power-efficient microcontroller fitted with a relatively capable dual-core CPU for multi-threaded operations such as high-throughput I/O. It also comes with built-in Bluetooth 4.2 and BLE support. Therefore, using external Bluetooth modules is not required. However, the onboard memory and storage are quite low and cannot sustain the kinds of data storage that are required in this project. Along with that, the community is relatively new and small compared to that of the previous devices. The community factor paired with a complex development environment makes it difficult for our use case.

After researching the specifications as well as the ecosystem that each of these devices has, it can be determined that the Raspberry Pi would be the most effective scanning device for this application. Due to it having a higher-end CPU, large memory and storage, and built-in radio, it can be used out of the box without any additional hardware requirements. The multiple OS support also comes in handy as it is much easier to develop the tools and programs as well as to execute them in a full-sized desktop environment.

3.5 Required Equipment

Once we have determined the main component of the experiment: the Bluetooth scanner device, we can create a final list of equipment that we will require. A list of required equipment and their purpose are as follows:

3.5.1 Raspberry Pi x 3

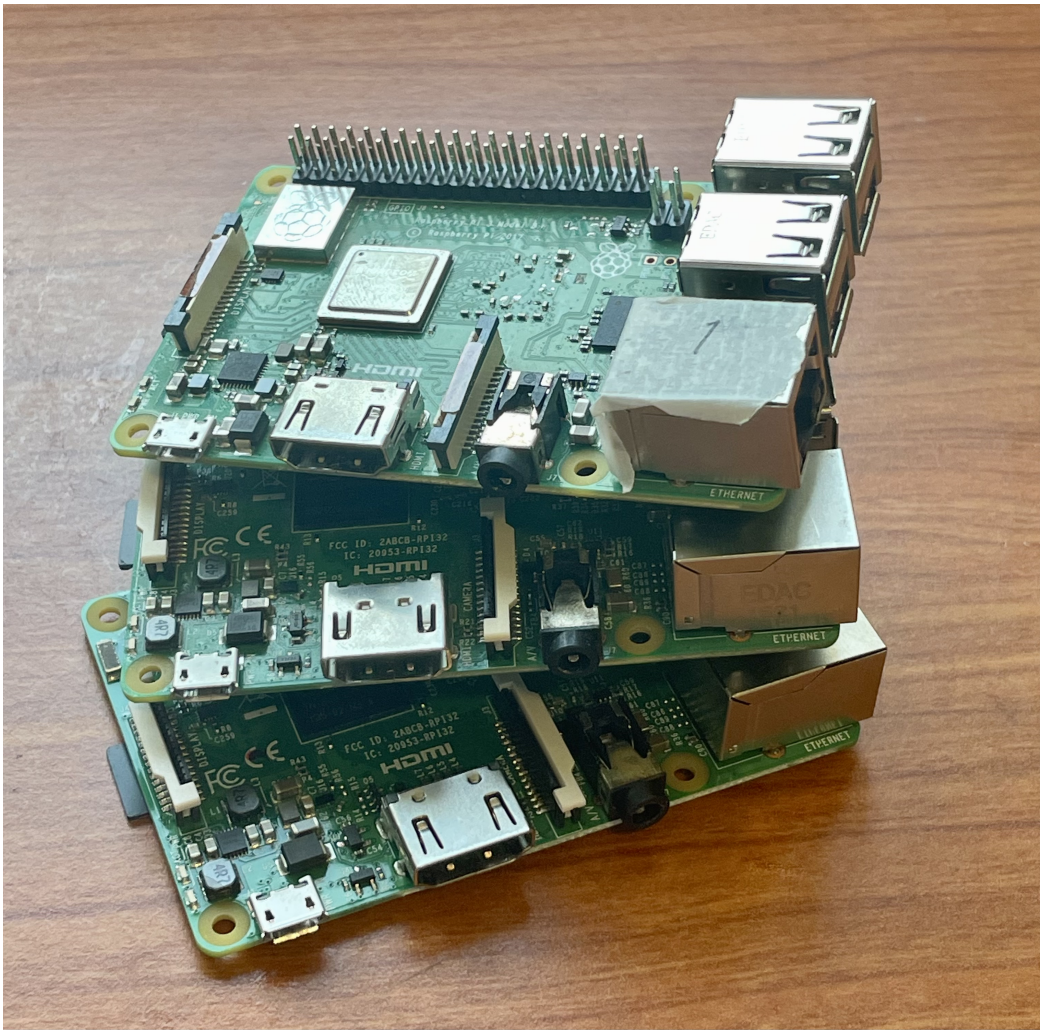

Figure 3.2: Raspberry Pi Model 3B+ x3

The Raspberry Pis act as the base stations in the experiment. A Raspberry Pi is an ARM-based microprocessor that runs its own Unix-like operating system based on Debian Linux (called the Raspberry Pi OS [26]). Moreover, Raspberry Pi OS comes preinstalled with several packages required during this experiment, and as a result, is the perfect fit. The microprocessor has a relatively average power requirement ranging from 6W to 12.5W (5v and 1.2A - 2.5A). However, it is important to note that power requirements are only necessary when the CPU is under full load and when the USB ports are all being utilized by peripherals. With a RAM of 1GB and an onboard SD card for storage (supports up to 64GB sized SD cards), it was found that the Raspberry Pi Model 3B+ is more than adequate for the purposes of this project.

3.5.2 Portable Batteries (Power Bank) x 3

In order to power the Raspberry Pi, a portable power source was required. Since the Raspberry Pis will not be connected to several USB peripherals, the maximum power draw in the context of this experiment would not be higher than 6W. Therefore, since modern power banks come with outputs of well over 15W and a relatively large capacity, a regular Power Bank was adequate for normal operations of the Raspberry Pi. It was finally decided to select a power bank with a maximum output wattage of 10W (5v and 2A) and a capacity of 5000 mAh (Milliampere-hours). That capacity would last well over 2 hours even at the Raspberry Pi's maximum power usage.

3.5.3 USB A to USB Micro B cable x 3

To connect the Power bank to the Raspberry Pi, a cable is required for each base station. It is important to ensure that the cable can indeed carry the required wattage as some lower specification cables can be limited to only 5W (5v over 1A)

3.5.4 SD Cards x 3

For onboard non-volatile memory as well as for loading operating systems, Raspberry Pis rely on SD cards that can be inserted into their SD card slot. As stated previously, they support up to 32 GB-sized SD cards. However, for the purpose of our experiments, 16-32GB is adequate.

3.5.5 Victim Device (Smart Watch)

To simulate a victim device, a smartwatch from an average company was chosen. The price was set at around 200 USD. By the end of the experiment, it is expected that the setup will be able to track this device in an outdoor environment up to a certain degree of accuracy.

3.5.6 LAN Cable

In order to connect to and configure the Raspberry Pi using a computer, a LAN cable is required. A Raspberry Pi can be connected to a computer and then SSH along with FTP protocols can be used to transfer files, executable code, and run commands on the Pi itself. Therefore, it becomes easier to develop the tools and code required for this project on a computer and then simply transfer it to the Raspberry Pi when needed.

3.6 Exploring Bettercap

According to their webpage [6], 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 IPv4/IPv6 networks*.". The bettercap project also implements a very flexible API for attack automation and data collection. For the context of our project, we will be using the Bluetooth scanning features provided by Bettercap.

Once installed and updated properly on the Raspberry Pi's OS, the bettercap package can be used in 2 major ways. The CLI tool can be easily accessed to use the full features of bettercap. Along with that, Bettercap also has a web UI, wherein a user can use a local bettercap server to monitor and interact with the WiFi and Bluetooth modules. Since the CLI implementation is quite limited in its automation features, it is unsuitable for use in the project. We will therefore be using the Web UI in order to read data from the Bluetooth module of the Raspberry Pi and finally save it to a CSV file on its onboard memory.

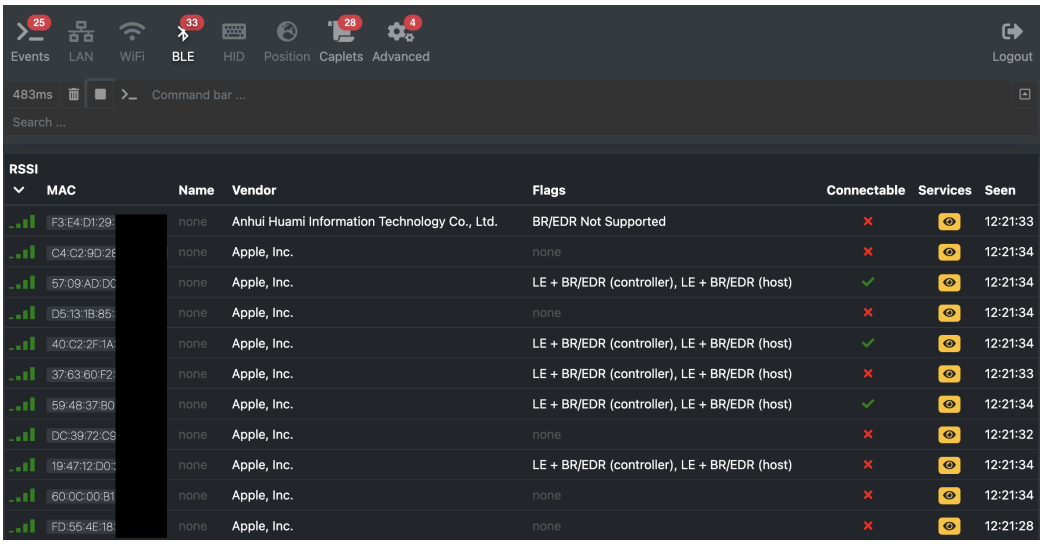

Figure 3.3: Web UI of Bettercap, in Bluetooth scanning mode

When bettercap is launched and its Bluetooth "recon mode" is turned on, it will poll the Bluetooth API and return several variables such as **RSSI**, **vendor name**, **detection time**, and so on. We can then write a program that can record the output and analyze it later.

3.7 Software Setup

To set up the software for the final experiment, we have to install NodeJS on the Raspberry Pis. This is to allow us to run a small NodeJS program on the Raspberry Pi. As discussed in Section 3.6, bettercap is used to scan for Bluetooth devices in the range of the Raspberry Pi. The NodeJS program runs on the Node runtime environment and essentially plugs into Bettercap's web UI and records each of the scanned devices along with identifying attributes. Since we were not CPU or power limited in this environment, NodeJS was a good option as asynchronous code execution allows for faster, non-blocking thread executions. At each refresh of the Bluetooth module, it has the potential to generate more than 30 data points, and as a result, a high throughput executable is well suited to reduce latency and backlog. Writing custom code for this also allows us to modify certain parameters to explore the features of the Bluetooth protocol stack inside the Raspberry Pi OS.

3.8 Hardware Setup

The hardware setup for this experiment is relatively simple. The protective cases are taken off of each Raspberry Pi, and they are turned on by connecting them to the power banks using USB cables. Afterwards, we connect to one of them at a time using a LAN cable and a laptop computer, log in via SSH, and start the scanning program for an appropriate amount of time. Then the LAN cable is disconnected and this process is repeated for all the Raspberry Pis.

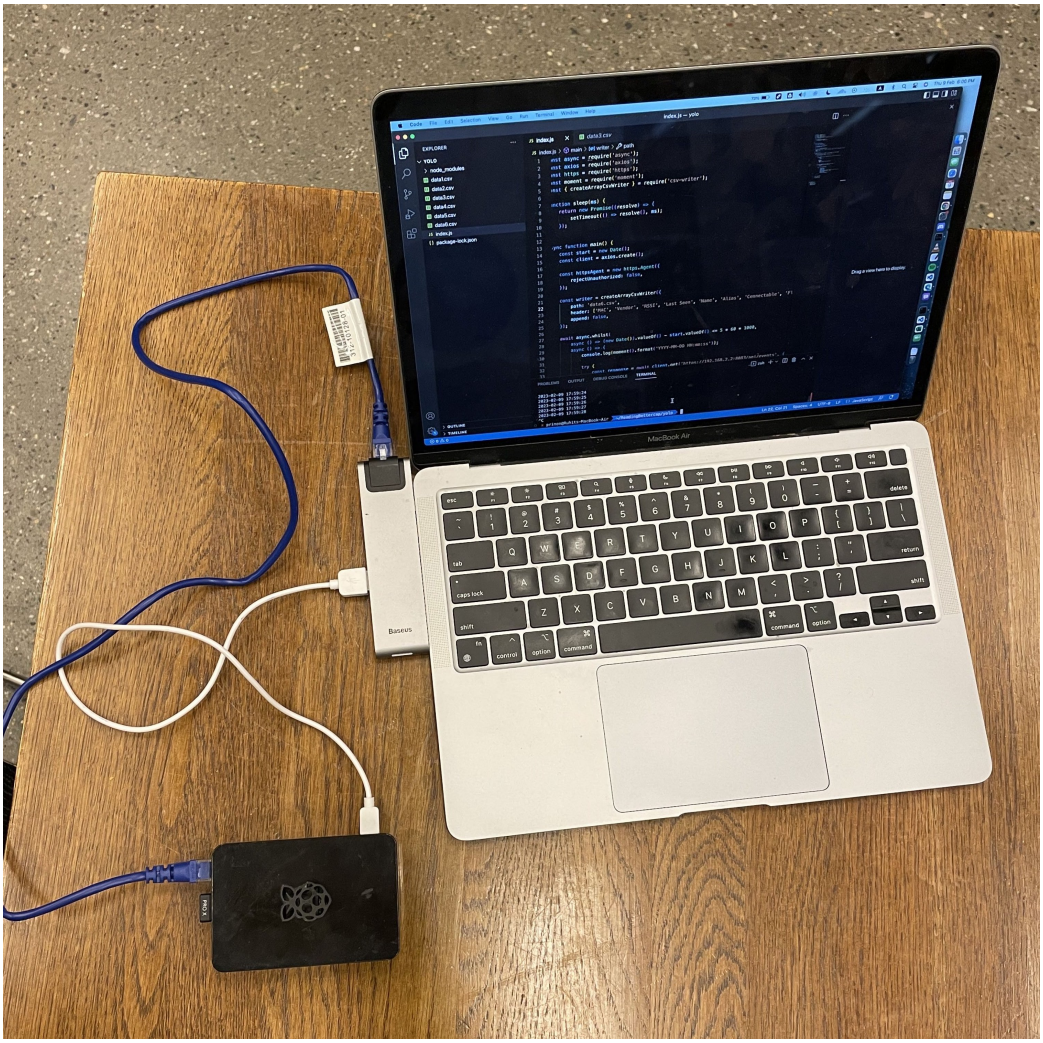

Figure 3.4: Portable Raspberry Pi connected to a PC via SSH

Chapter 4

Experimentation

This chapter focuses on several crucial on-field experiments with the hardware as well as some of their findings. It concludes with the design of a final experiment that aims to track a subject(victim) in a 2-dimensional open space using their on-body IoT device

4.1 Pre-Experiments and Hardware Calibration

To perform a successful experiment, we must first understand a few things about our equipment as well as the characteristics in a real-world environment. Therefore, 4 different experiments were designed to determine the parameters and setup of the final experiment.

4.1.1 Case Test: Does a Plastic Case Hinder RSSI Performance

By default, the Raspberry Pi comes with a plastic cover that protects the board and its pins from the environment. However, it is hypothesized that the plastic cover can, to a noticeable degree, block Bluetooth-carrying microwaves. The purpose of this experiment is to understand to what degree the case hinders RSSI readings.

To run this experiment, 2 arbitrary distances were chosen, 2 meters and 6 meters, both of which should easily fall inside the radius of Bluetooth's range. It was also performed in a large open space with minimal Bluetooth devices available to reduce interference. Then the Raspberry Pi was placed with the case and RSSI readings were taken for 30 seconds at both distances. The same setup was repeated without the case and the readings were collected.

Table 4.1: Case Test Data

	Average RSSI	ST Deviation
With Case	-70.737	5.478
Without Case	-66.938	3.388

Looking at the RSSI values in Table 4.1, it is clear to notice that the case indeed has a significant effect on not only the RSSI values itself, but also the variance of the values. Therefore, the conclusion is that from here on out, the Raspberry Pi should be used without a case to maximise accuracy.

4.1.2 Orientation Test

According to Vincent Gao [16], even the layout of the internal circuits of a Bluetooth antenna can cause discrepancies in the signal being received. This can be caused by electromagnetic interference as well as plastics and metals blocking the microwaves from reaching the antenna effectively. The purpose of this experiment is, therefore, to find whether there is a difference in RSSI between one side of the Raspberry Pi and the others provided all other factors remain unchanged.

Similar to the first experiment, this one was also performed at 2m apart, and again at 6m apart. One of the Raspberry Pis and the Victim device were placed at the aforementioned distance apart, and the Raspberry Pi was set to Bluetooth Scan mode. The data was recorded for 30 seconds, and then the Raspberry Pi was carefully rotated 90 degrees to face the victim device from a different side. This was repeated for each side and it was repeated for both distances. It was also repeated using another Raspberry Pi in order to account for per-device discrepancies.

Afterwards, the data was compiled and sorted so that only the RSSI values of the victim device were kept. For each state of the experiment, the mean RSSI value as well as the standard deviation was calculated and compared.

Table 4.2: Orientation Test Data

Sides	**Average RSSI**	**Standard Deviation**
East Facing	-63.9688	3.1410
North Facing	-64.3125	2.8039
West Facing	-65.4688	3.1285
South Facing	-72.0000	4.9624

From the data in Table 4.2, it can be observed that one side of the Raspberry Pi does indeed have marginally lower RSSI values at the same distance which indicates that the device has a "dead" side which is less sensitive to Bluetooth signals. More crucially, it was observed from the data that the standard deviation of the less sensitive side is much higher, which means that any readings taken from that side will have a higher margin of error. It can also be observed that there is not much of a difference between the two Raspberry Pis, so they can be thought of as identical devices when it comes to circuit layouts. It is therefore understood that the "dead" or less sensitive side of the Raspberry Pi should be avoided as much as possible during the course of the project.

4.1.3 Relationship between RSSI and Distance

Since we are using RSSI to estimate the distance, we must first build an understanding of the relationship between these two variables. Therefore, an experiment is proposed and its purpose is to find the accuracy of RSSI as a distance indicator, and to find the limits of RSSI as said indicator. The results of this experiment will also be used to create a lookup table for the final experiment. This experiment was performed in a similar environment to that of the previous one. Moreover, it was made sure that the "dead" side of the Raspberry Pi was not facing the victim device.

In order to perform this experiment, one Raspberry Pi was set next to the victim device at a known and measured distance, and the scanner was started for 30 seconds. For 30 seconds, the data was recorded and then the scanner was stopped. The experiment was repeated at 1-meter intervals starting from 0m(devices next to each other) to up to 12 meters.

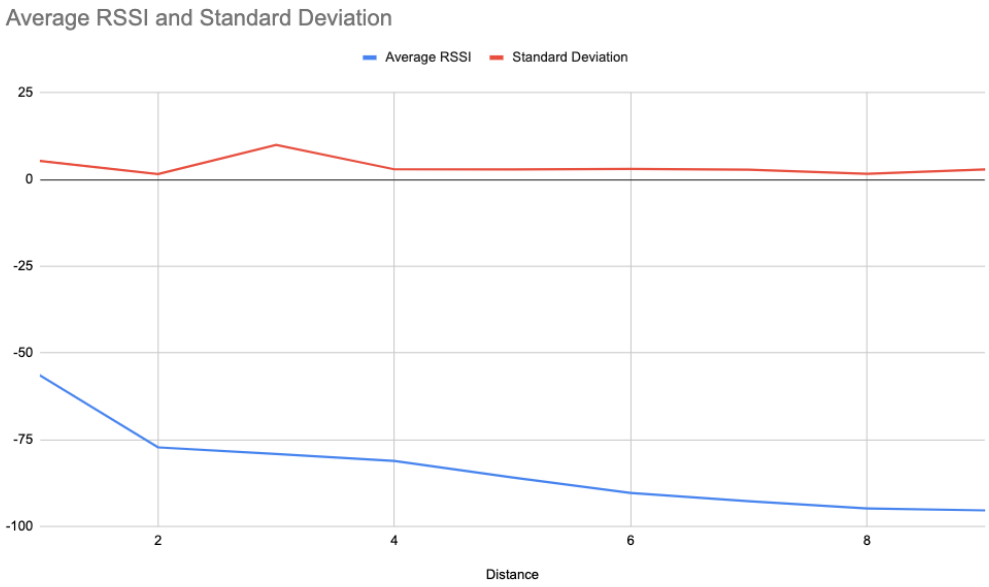

Figure 4.1: Average RSSI and Standard Deviation with respect to Distance

Finally, the data was collected, and sorted similarly to the previous experiment and the mean RSSI for each known distance was calculated along with the standard deviation. The standard deviation for each data point was observed to be less than 10, which leads us to conclude that RSSI is indeed a good enough indicator for distance. Therefore, given the RSSI of the victim device at a given time, we can estimate the distance using this lookup table that was created in this experiment. The data collected in this section will be used further in Section 5.4 to create a learning algorithm to further improve the estimates.

4.1.4 Bluetooth Scanning Frequencies: Where is the Point of Diminishing Returns?

As explained by Faragher et al. [14], there is a certain point where increasing the frequency of updates from the Bluetooth scanner results in diminishing returns: a point where the results do not change even if we increase the frequency (Note: This term "frequency" here describes the rate at which data is read from the Bluetooth module and has nothing to do

with the microwave frequency at which Bluetooth signals are transmitted). In the authors' testing, it was found that there were no improvements in readings above 10Hz. In order to test it for our set of hardware and software, an experiment was designed. The purpose of this experiment is to find the practical limits of Bluetooth's scanning refresh rate.

In this experiment, we set up a base station and a victim similar to that of the previous ones, but we vary the frequency at which data is saved. The refresh rate can be trivially modified by editing the NodeJS code in Section 3.2. 5 different refresh rates will be tested in this experiment, 10Hz, 5Hz, 2Hz, 1Hz and 0.5Hz. For this experiment, we will have a moving victim to simulate changing RSSI. Therefore, the victim is instructed to walk along a predetermined path with the distance between the base station and the victim varying going from 1m to 10m and then back to 1m.

To analyze the data, we can try to find how many duplicate RSSI values are amongst adjacent readings (adjacent in chronological order). Since RSSI should be varying constantly, higher duplicate readings mean that we are getting overlapping readings from the Bluetooth scanner, and overlapping results are essential of no use to us. From the data, it can be observed that frequencies above 2Hz hardly result in any improvements as most of the data can still be found on lower frequencies. Therefore, scan frequencies above 2Hz for this set of hardware and software give diminishing returns.

4.2 Final Experimentation

Following the series of experiments, we can make assumptions and start with the final experiment where we hope to track the precise location of a victim leveraging their IoT device in a 2-dimensional environment.

4.2.1 Performing The Experiment: Considerations and Assumptions

To reiterate what we have learned from past work and present experimentation, we are keeping a few assumptions in mind for the final procedure:

- All the protective cases were taken off of the Raspberry Pis
- The dead zone on the Raspberry Pi was avoided as much as practically possible
- A scan frequency of 2Hz was chosen

4.2.2 Procedure Setup

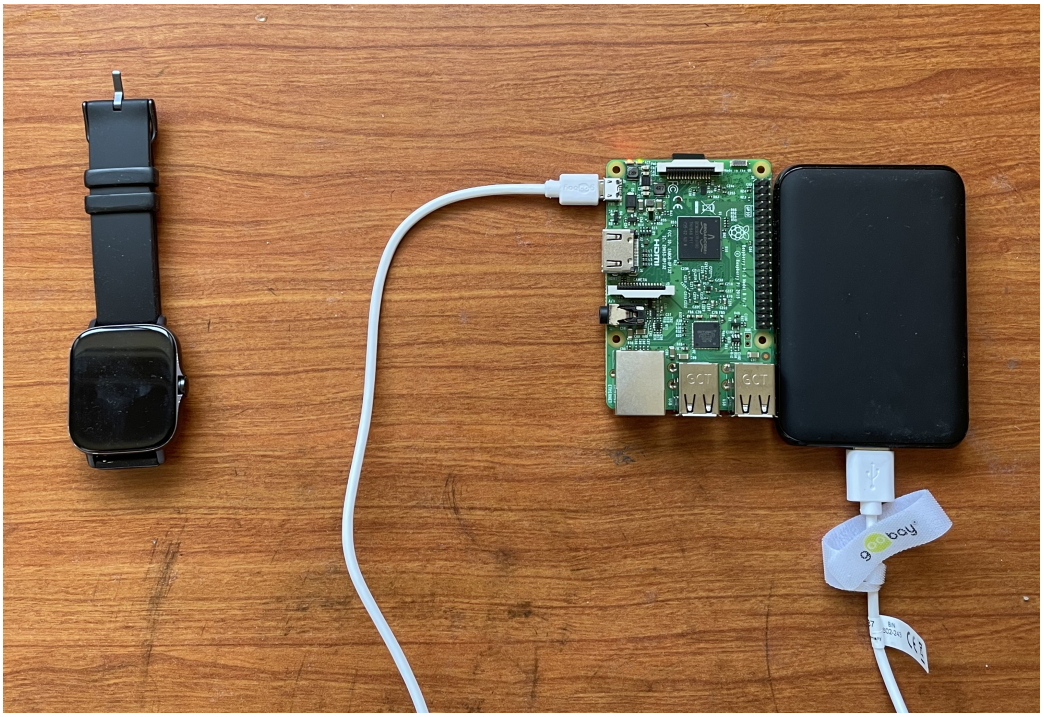

Figure 4.2: Portable Powered Raspberry Pi With Victim Device

Following the project description and pre-experimentation, the hardware was set up most optimally according to the data and knowledge learned. For the final experiment, 3 base stations were set up 7.5m apart from each other creating a right-angled triangle. A victim was given a smartwatch to wear on their wrist. Then, the base stations were set up in known positions around a field.

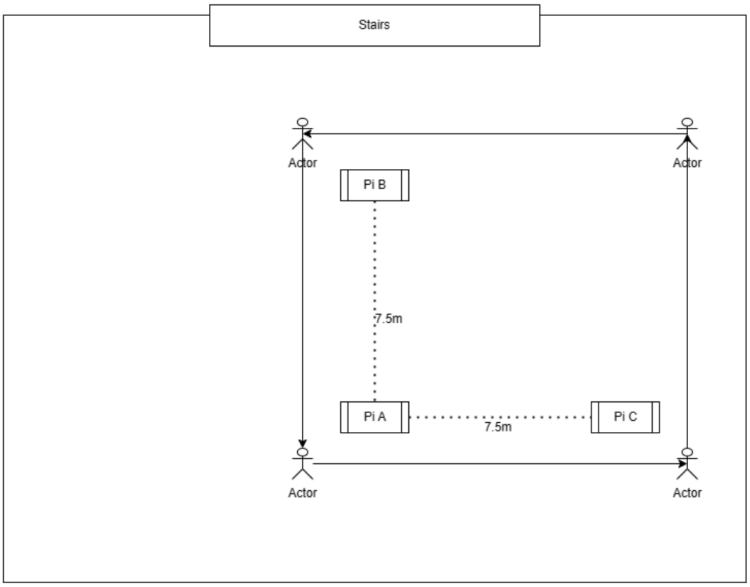

Figure 4.3: Final Procedure Setup

Their GPS Coordinates were taken using an app for iPhone named "My GPS Coordinates" which provides GPS coordinates with up to 1m accuracy in ideal conditions. Finally, the base stations were set to scan mode, and the victim was instructed to walk on a pre-determined path that makes a distinct and discernible shape if traced properly. When the victim had finished walking the path, the scans were turned off and the generated files were copied from each file onto a computer using FTP and SSH to be analysed.

In order to ensure that the environment does not have a large effect on the data, the experiment was repeated in 3 different environments:

- A large auditorium with benches and a ceiling
- A large and open plaza with some obstacles but mostly open-air in between the scanners and the victim
- A grass field with no obstacles

Chapter 5

Data Analysis

In this chapter, it is explained how the data collected in experimentation are processed and analysed, and what techniques are used to analyse the data.

5.1 Data Description

During the final experiment, the Raspberry Pis collect data and store them in CSV (Comma Separated Value) files. The data captured by the Raspberry Pi includes all the devices that are in its range, and with every update a newer set of data is generated and written to the files. Here is a list of each of the variables we record into the CSV file at every interval:

- RSSI
- Vendor
- MAC
- Last Seen
- Flags

Therefore, if we have to analyse this data, we must first filter out the unwanted data points and sort them in a proper way. Below is a sample image of a subset of the raw data captured by one of the Raspberry Pis. As discussed previously, we are capturing 5 data points for each device during each refresh:

MAC	Vendor	RSSI	Last Seen	Flags
F3:E4:D1:29:	Anhui Huami Information Technology Co., Ltd.	-67	2023-03-14 18:3	BR/EDR Not Supported
F7:28:0A:C3:	Apple, Inc.	-52	2023-03-14 18:3	
CE:45:00:11:	Apple, Inc.	-89	2023-03-14 18:3	
D6:EB:3D:BF	Sony Corporation	-74	2023-03-14 18:3	
E0:54:58:F5:4	Apple, Inc.	-71	2023-03-14 18:3	
63:A8:73:6D:	Microsoft	-89	2023-03-14 18:3	
73:9B:BB:9F:	Microsoft	-84	2023-03-14 18:3	
65:07:6B:BC:	Microsoft	-85	2023-03-14 18:3	
DD:34:8F:E3:	Apple, Inc.	-89	2023-03-14 18:3	
6E:9A:9B:46:	Sony Home Entertainment&Sound Products Inc	-89	2023-03-14 18:3	
15:FD:02:C5:	Apple, Inc.	-80	2023-03-14 18:3	
70:BA:12:1F:7	Sony Corporation	-84	2023-03-14 18:3	
76:7B:B3:D6:	Sony Corporation	-89	2023-03-14 18:3	
7E:E3:1F:ED:	Sony Home Entertainment&Sound Products Inc	-84	2023-03-14 18:3	
F7:80:C8:9B	Apple, Inc.	-69	2023-03-14 18:3	

Figure 5.1: Sample of raw CSV data collected from the Raspis

5.2 Dataflow Diagram

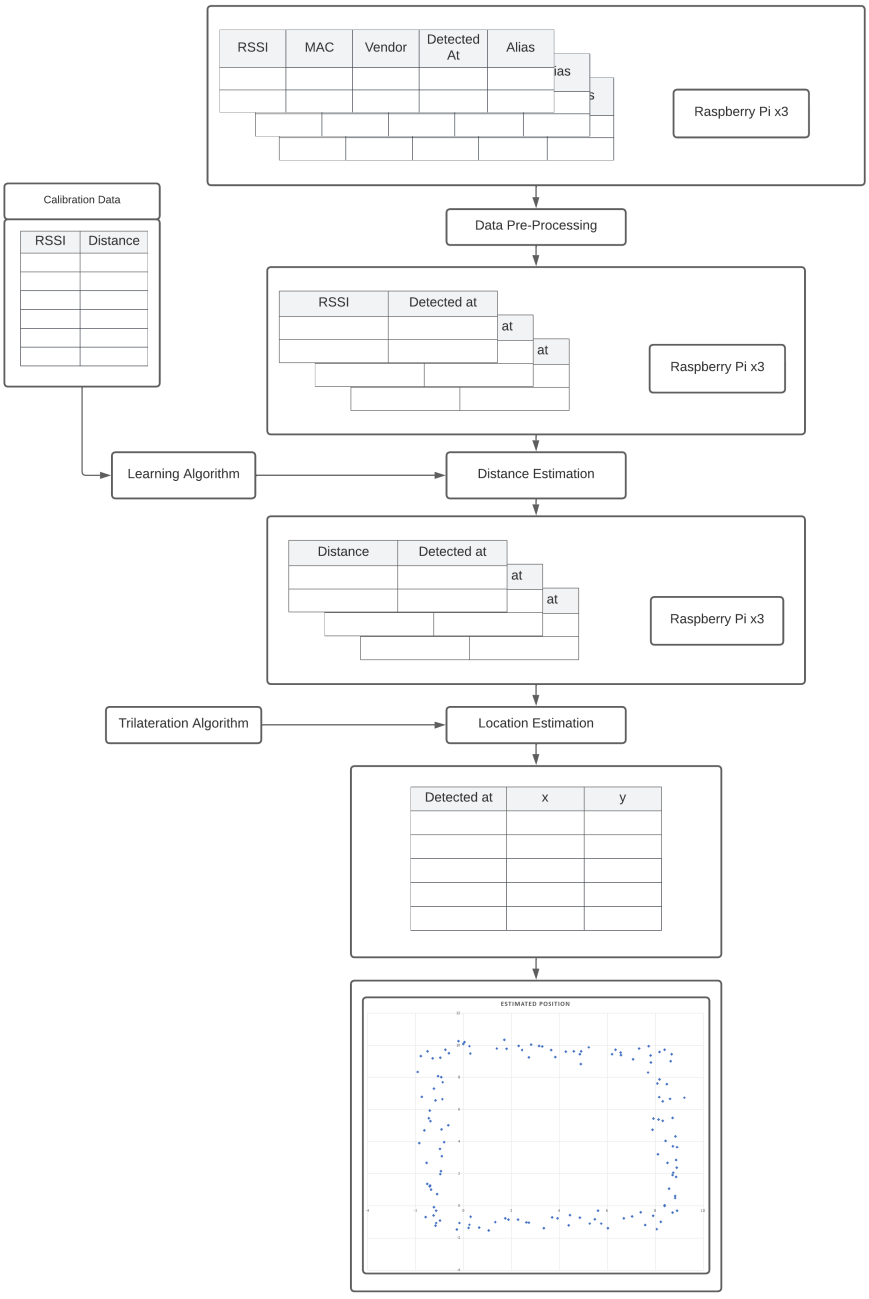

Figure 5.2: Dataflow diagram of the final experiment

Above is the dataflow diagram that describes both the data and the processes involved in the analysis of the experiment findings. First, we reduce the data's width by filtering, sorting, and dropping unnecessary data. Then we use the learning algorithm discussed in Section 5.4 to replace the RSSI values with distance values. Then we take the distance

values and use a trilateration algorithm to estimate the position of the victim device. We can finally use this position data to graph exact positions of the victim. Each of the steps will be discussed in further detail below.

5.3 Data Pre-processing

In order to analyse the behaviour of a single device, we must first filter out all the other devices from the scanned files. In our case, we are targeting a victim device that has a Vendor name: "Anhui Huami Information Technology Co., Ltd" and a MAC address of "F3:E4:D1:29:xx:xx". Therefore, in order to clear out all the other devices, we can simply filter out all data rows other than this device.

Moving on, we can sort the rows according to the Last Seen values, going from earliest to latest detection. We will also trim the rows according to the start time and end time noted during the experiment. Following that, we will not require the Flags, MAC and Vendor column, as it only contains data from one device and identification is no longer required. Therefore, the aforementioned columns are dropped as well, leaving only RSSI and Last Seen columns. The whole process is then repeated for each of the CSV files from the Raspberry Pis.

5.4 Learning Algorithm for Distance Estimation

In the experiment explained in section .... we recorded device RSSI and the corresponding distance values. This made it possible for us to to represent the relationship between RSSI and distance. However, even though the discreet points give us some points of reference, in order to get more accurate results from our setup, it is important that we essentially "fill in the gaps" between the data points. Therefore, a learning algorithm is required to find more continuous approximations of distance.

Let us assume that distance is the independent variable x, and RSSI is the dependent variable. We can then plot the x and y points on a Cartesian axis.

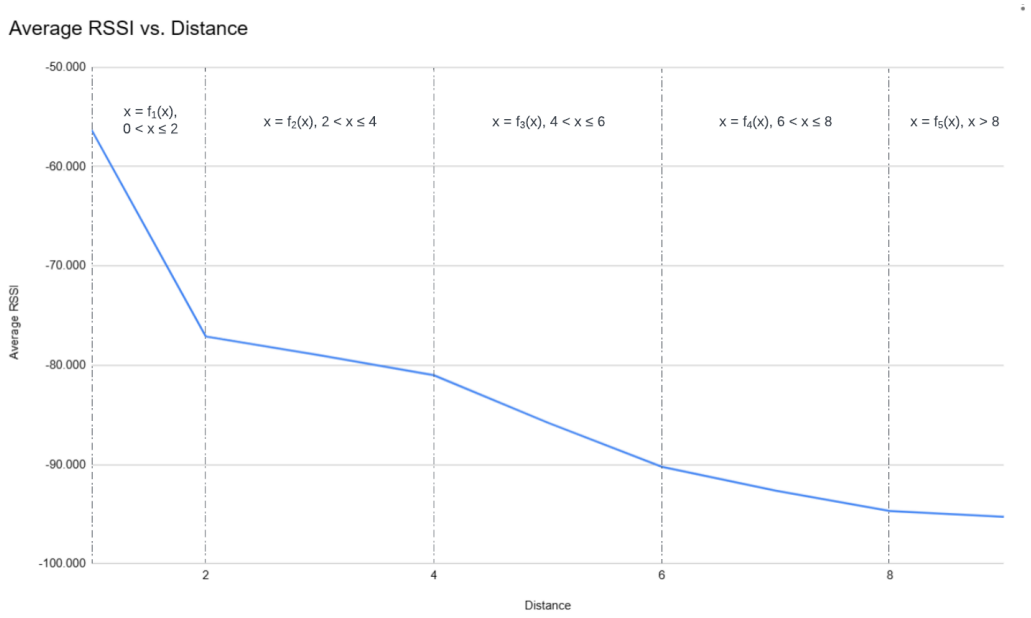

Figure 5.3: RSSI - Distance calibration data forms a Piecewise function

Here, we can clearly see that the function resembles a piecewise function. There are distinct anchor points in the curve that represent a shift in the relationship that closely characterizes a piecewise function f(x) where

$$f(x) = \begin{cases} f_1(x) & \text{if } 0 < x2 \\ f_2(x) & \text{if } 2 < x4 \\ f_3(x) & \text{if } 4 < x6 \\ f_4(x) & \text{if } 6 < x8 \\ f_5(x) & \text{if } x > 8 \end{cases} \tag{5.1}$$

Linear interpolation and polynomial regression are both methods used for approximating a function from a set of discrete data points.

Linear interpolation is a simple method for approximating a function that assumes that the function is a straight line between each pair of adjacent data points. This method works well when the function is relatively simple and can be well approximated by a straight line. Linear interpolation can be a good choice when you only have a small number of data points and need to quickly estimate the value of the function at a specific point within the range of the data.

On the other hand, polynomial regression is a method for approximating a function using a polynomial of a certain degree. This method works

well when the function is more complex and cannot be well approximated by a straight line. Polynomial regression can provide a more accurate approximation of the function when there are many data points and the function has a higher degree of complexity.

Since the relationship between distance and RSSI is relatively more complex and cannot be accurately approximated by a straight line, polynomial regression is preferred. In particular, the signal strength can be influenced by various factors such as environmental conditions, signal interference, and hardware variations. Therefore, a higher degree polynomial may be required to capture the nonlinear relationship between distance and RSSI.

However, it's important to keep in mind that polynomial regression can be prone to overfitting when the degree of the polynomial is too high. Overfitting occurs when the polynomial is too closely tailored to the data points used for regression, resulting in poor performance on new data points. Therefore, after we are finished with interpolation, it is important to visually inspect the generated data points in order to ensure that the regression algorithm has not overfitted for the original data.

5.4.1 Spline Interpolation

Spline interpolation, or more specifically in our case Polynomial Spline Interpolation is a mathematical analysis tool that builds upon the previously discussed Polynomial regression. It is most appropriate for interpolating data points wherein the function is a piecewise function with each piece being fitted to a distinct polynomial function. It essentially assumes that at certain points on the x-axis (known as knots) the function changes to a different polynomial function. For our case, it is well suited in order to interpolate more data points as we can identify the "knots" of this function at x = 2, 4, 6, 8 and so on. Therefore we will implement a spline function that will take any value of x (distance) within the range $0<x<12$ and return the value of RSSI at much more smaller intervals.

To implement the spline function into a program, we can use the library *spline-js* and the NodeJS runtime environment. The library allows us to generate a spline function and generate values of y given values of x. However, in our final experimentation, we have recorded RSSI values and we hope to estimate the distance from that RSSI value. This would mean that we want to estimate x given y, which is not the natural direction at which a spline function operates. Therefore, when we have to use the "invert" function to estimate a point x given y. The actual implementation of this function will be discussed further in Section .....

5.4.2 Spline Interpolated Distance Estimation

After the data has been cleaned up, the RSSI can finally be replaced with the corresponding distance value. To do so, spline interpolation will be used in conjunction with the data generated from the distance calibration experiments. Below is a pseudocode of the actual program used to generate a spline and to replace the RSSI values its with distance equivalent.

Algorithm 1 function DistanceEstimation

DistanceEstimation

Import spline-js
reader(file) ← Import CSVReader

x[] ← x values of the calibration data
y[] ← y values of the calibration data

splineFunction = spline-js.getNaturalSpline(*x*, *y*)

RSSI[] ← RSSI Values
Distance[RSSI.length]
c ← 0

WHILE(**c<RSSI.length)**
    Distance[c] = spline.invert(splineFunction, RSSI[c])

WriteToCSV(Distance)

To create the spline function, we use the *getNaturalSpline* function from the *spline-js* library. This function generates a cubic spline, which is a mathematical representation of a smooth curve that passes through the given set of points. This spline function can be used to interpolate or extrapolate new values along the curve.

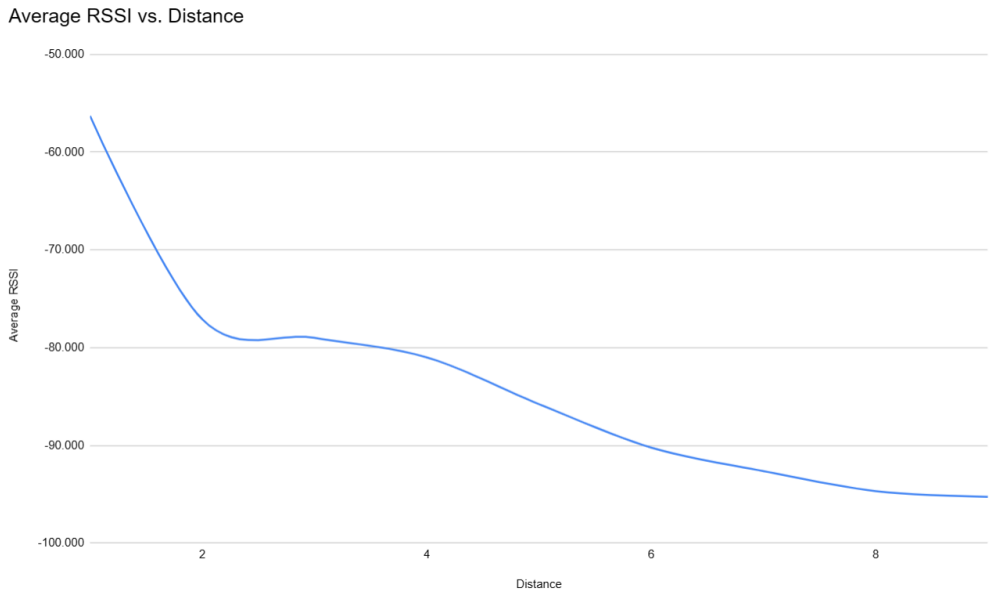

Figure 5.4: Spline interpolation of the RSSI - Distance relationship

If we visualise the generated Spline, we can observe how the graph changes from a set of several lines to a more smooth curve.

After creating the spline, the *invert* function is used to convert RSSI values to corresponding distance values. The invert function takes RSSI values of the victim device and outputs an approximate distance value by looking at the curve generated in Figure 5.4. The invert function is applied to each RSSI reading from all the Raspberry Pis.

Finally, the we write the calculated distance values back to the CSV file for further processing.

5.5 Triangulation - Triliteration Algorithm

In the field of positioning systems, trilateration is a commonly used method to estimate the position of a device based on distance measurements from fixed reference points. The last step of analysing the data is to use 3 distance values to estimate the position of our victim device at each point in time. To do so, we must first simplify the trilateration equation and write a program that can run the algorithm.

Several studies have used trilateration algorithms to estimate the position of devices in indoor environments. For example, in a study by

Liu et al. [30], a trilateration algorithm was used to estimate the position of a mobile device based on Bluetooth signals from fixed beacons. Similarly, in a study by Si et al. [8], a trilateration algorithm was used to estimate the position of a mobile robot in a warehouse based on signals from fixed WiFi access points.

$$(x - x_1)^2 + (y - y_1)^2 = d_1^2 \tag{5.2}$$

$$(x - x_2)^2 + (y - y_2)^2 = d_2^2 \tag{5.3}$$

$$(x - x_3)^2 + (y - y_3)^2 = d_3^2 \tag{5.4}$$

The trilateration algorithm uses the distances between a point and three fixed reference points in a Cartesian plane to estimate the position of the point. The relationship can be explained by the equations 5.2, 5.3 and 5.3. Here, (x, y) is the coordinates of the victim device and (x_1, y_1), (x_2, y_2), and (x_3, y_3) are the 3 fixed reference points. d_1, d_2, and d_3 are the distances between the victim device and the 3 reference points.

In our case, the fixed reference points are the base stations A, B, and C, with relative coordinates of A(0,0), B(0,7.5), and C(7.5,0) in a Cartesian plane representing the surroundings. Each unit in the coordinate plane represents a distance of 1 meter.

We can simplify the equations by replacing the know coordinates of the points A, B and C. The simplified trilateration equation for our case can be expressed as:

$$(x)^2 + (y)^2 = d_1^2 \tag{5.5}$$

$$(x)^2 + (y - 7.5)^2 = d_2^2 \tag{5.6}$$

$$(x - 7.5)^2 + (y)^2 = d_3^2 \tag{5.7}$$

Further simplifying, we can get a set of equations that can be evaluated to find the estimated coordinates of the victim device:

$$x = (d_1^2 - d_2^2 + 56.25)/15 \tag{5.8}$$

$$y = (d_1^2 - d_3^2 + 56.25)/15 \tag{5.9}$$

5.6 Location Estimation

Using the above equations, we wrote a program to estimate the x and y coordinates of the victim device at each given point. It is important to note that the x and y coordinates obtained are not the exact GPS coordinates of the victim device, but rather a relativistic representation of its position.

To obtain the exact GPS coordinates of the victim device, we can use the GPS coordinates of the base stations as reference points. By applying basic trigonometry and coordinate transformations, the exact GPS position of the victim device can be calculated with high accuracy.

If we run the program and input all of the recorded distances, we can find a set of x and y coordinates. These coordinates can be then used by any visualisation tool to show where the victim device was detected during the experiment.

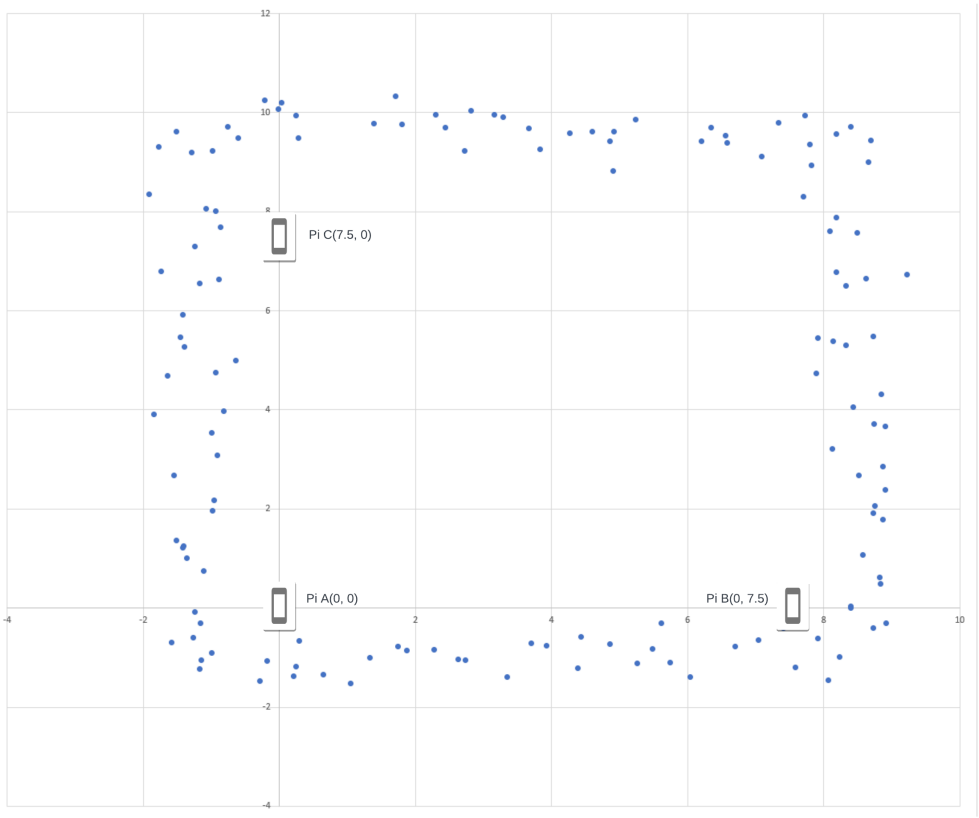

Figure 5.5: Scatter Chart showing the relative locations where the victim was detected

If we take the x and y coordinates to generate a scatter plot, we can visualise the detected points. Looking at the scatter plot above, we can see that there is a clear representation of the path that the victim had walked as shown in Figure 4.3 to a high degree of accuracy. The points represent an approximate path the victim had taken throughout the final experiment. We can finally conclude that the overall accuracy of using a calibrated system to triangulate a victim IoT device is quite high.

5.7 Angle and Vector Estimation

In a real-world environment, it might be easier to visualise angles and vectors instead of points on an imaginary 2D plane. Along with that, it is easy to visually confirm which device is being detected by simply looking in a specific direction. This would potentially make it easier to "spy" on victims in specific buildings, specific directions or cars for example. In order to convert x and y coordinates to angles and distances, we can convert the Cartesian coordinates to Polar coordinates. The relationship is represented by the equations:

$$x = r\cos\theta \tag{5.10}$$

$$y = r\sin\theta \tag{5.11}$$

where r represents the distance from the origin to the point, and θ represents the angle. To solve for r and θ, we can rearrange the equations:

$$r = \sqrt{(x^2 + y^2)} \tag{5.12}$$

$$\theta = \arctan\left(\frac{y}{x}\right) \tag{5.13}$$

Equations 5.12 and 5.13 can then be executed using the same program to determine the angle and distance at which the victim was located. It is important to note that the angle and distance will be relative to the scanner at the origin point, Pi A (0, 0).

5.8 Accuracy

In order to quantify the accuracy of the results, a reference is required. To do so, we can plot the actual shape the victim had taken during the experiment. This line is represented by the red line in Figure 5.6. The

square is built using the accurate measurements that were taken during the experiment design and experiment execution.

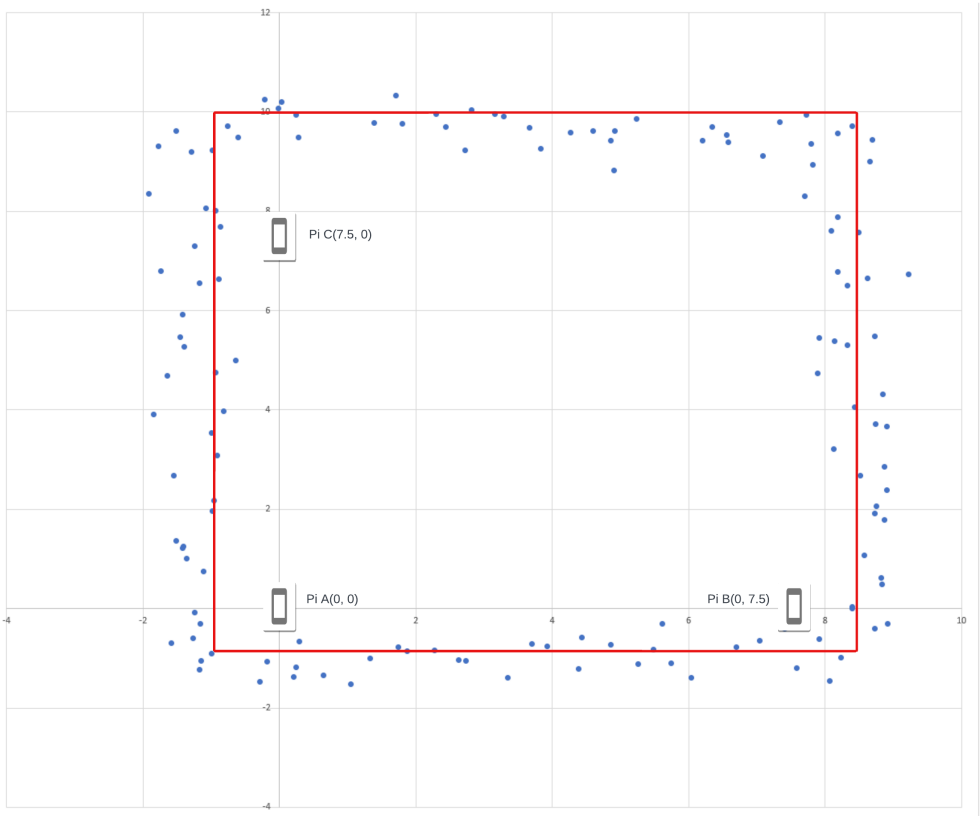

Figure 5.6: Resultant scatter chart with reference line

With a reference line in place, the accuracy of the results can be measured. In order to compare points to points, coordinates can be generated corresponding to the number of points that exist in the scatter chart. The red line represents where the device was theoretically located. Then the distance between each of the reference points and the experimental points can be calculated to find the accuracy of the readings of each point. To calculate the distances between these points, we use the equation:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{5.14}$$

Once the distances are calculated, it is found that the average distance between the reference point and the equation is 0.437m with a standard deviation of 0.223. Therefore, the location of the victim can be determined with an accuracy of up to <0.5m.

Chapter 6

Discussion

This chapter discusses the results of the experiment and the implications of the results. The chapter also presents the study limitations and potential future work paths.

6.1 Results

The system that was developed is able to distinguish unique users using their wearable devices with an accuracy of less than 0.5m with a sphere of detection of over $12m^2$. This level of accuracy is enough to gain harmful insights into a victim such as spying on their home layout and in-home devices. It can be used to tell if a victim is home, and can be used by employers to track the movements and activities of an employee without their knowledge. The implications of these results raise questions about the legality of such tracking, and whether it constitutes a violation of individual privacy rights.

6.2 Future Work

However, there is much work to be done in the field of privacy and security of IoT devices. More specifically, it is feasible to build a real-time tracking system that can visualise the environment using real-time communication modes that are supported by Raspberry Pis. A natural next step to this process is to build a tracking system that can track devices in real-time and extract identity attributes at the same time. For instance, the Raspberry Pis could use the WiFi modules to communicate with each other and relay the scanned data in real-time to one another and calculate the estimated location of all devices in real-time. Albeit slightly more complex and requiring more computing power, it is possible given the state of current technology.

Along with that, it is also feasible to create a 3-dimensional tracking system with the inclusion of an extra 4th Bluetooth scanning device. This would also increase the computational complexity. Moreover, stronger Bluetooth antennas could be used to create a better estimation model and build a more sensitive system.

6.3 Privacy Implications

Using an openly broadcast metric on the Bluetooth channels, we were consistently able to find the precise location of a victim's device without their consent. As explained in the literature referred to in this paper [29][21][3][14][20], this has been proven many times by using different scanner devices, different victims, different antenna designs, as well as different algorithms and data processing techniques. Regardless of the different approaches of the researchers, the fact remains that it is quite possible to create a system to track a user's movements by targeting their wearable devices.

Being able to track a person using their smartwatch raises significant concerns about privacy and security. Smartwatches are often used to monitor personal health and fitness data, and they may also contain sensitive personal information such as financial data or contact lists. If this data falls into the wrong hands, it could be used for malicious purposes such as identity theft, stalking, or blackmail. Moreover, the ability to track a person's location and movements using their smartwatch raises significant ethical concerns. It could be used to monitor the movements of employees, children, or vulnerable individuals, without their consent or knowledge.

In a paper by Ching et al, [10], the researchers analysed 3 wearable devices and found all of them had several potential attack vectors that could be used by malicious users to gain unwarranted access. Amongst the devices there were 2 devices (Google Glass and Fitbit Fitness trackers), they also tested attack vectors that could be easily combined with the technique discussed in this paper to create a multi-vector attack to compromise the user's location, privacy, and availability.

A study into the awareness of privacy and security threats by Bellekens et al. [5] shows us the clear gap in knowledge of consumers concerning products they own as well as technologies they rely upon. In their research, the authors conducted a survey that revealed a concerning pattern of ignorance among most of the participants, indicating a lack of awareness of security and privacy threats. The results of this survey illustrate the widespread misinformation or lack of information among the general population regarding the seriousness of security risks associated with IoT devices. Further research in this field by Khan et al. [18] has also shown that many consumers fail to take adequate security measures to protect their IoT devices, making them vulnerable to attacks. To address this issue, it is crucial to provide education and awareness campaigns to the general public on the importance of protecting their privacy and security while using IoT devices.

6.4 Threat Prevention and Mitigation

Seeing as Bluetooth is a relatively old technology with years of research and development in threat prevention and mitigation, it is a relatively safe protocol for wireless communication. However, the biggest threat to Bluetooth devices is the implementation of the Bluetooth standard. It is quite common for vendors to not focus on the correct implementation of the protocol which leads to successful attack vectors arising. These issues

can be addressed by stronger regulations around the globe such as GDPR and CCPA.

Even though GDPR Article 25 [11] requires that data controllers implement privacy by design and default principles in the development and deployment of Bluetooth and IoT devices, this rule seems to not extend to RSSI broadcasts. To mitigate this threat, the Bluetooth standard can limit the use of RSSI as a measure of proximity and signal strength. Currently, the RSSI readings are in between a range of 0 to -120, which is precise data that is being broadcast. This data could be obfuscated by masking the RSSI readings into 4 categories: Strong, Normal, Weak and Lossy, going from the highest RSSI to the lowest. Only this categorical reading should be available to high-level users or data controllers. It should be mentioned that this obfuscation should only be implemented on IoT and personal devices. Since Bluetooth is used in commercial applications such as asset tracking in factories and person tracking in airports, for example, more precise RSSI readings are crucial for such applications.

6.5 Limitations

As with any practical study, there are several limitations of this study that can be further improved upon by other studies.

For instance, this study only focuses on the detection and localisation of IoT devices in a 2-dimensional field. Our experimentation assumes that the victim is always at the same height as the scanners, which is not always the case in real-world environments. There are cities and neighbourhoods with a significant z-height difference between the 2 positions, and the victim can also be on different floors of a building. The equations used in this study will fail when presented with data from situations where the victim is in a different z-plane. One potential solution to this limitation is to employ an additional Bluetooth scanner that can resolve the height of the victim. However, this requires the use of a more complex version of the trilateration equation, involving 4 equations and 4 sets of variables. Implementing this solution will result in a more complex algorithm, requiring significant computing power to resolve in real time. Additionally, visualizing 3D data in such scenarios is a challenging problem that requires further exploration.

One other limitation of the hardware used in this setup is the Bluetooth module in the Raspberry Pis. The in-built Bluetooth transceiver is built into the motherboard itself and does not have a large antenna. This

results in a lower sensitivity to microwave frequencies and may cause inaccuracies in the readings. The built-in transceiver is clearly good enough as a proof-of-concept, but in order to improve on the results, improvements can be made to the system setup. One solution to this problem is to use external Bluetooth antennas that can boost the sensitivity of the system. This should ideally result in more accurate RSSI readings and eventually a more precise and accurate location-sensing system.

Another limitation of the procedure described in this project is the range limitation. Since we are relying on 3 data points from 3 different Raspberry Pis, it is crucial that the victim device has to be detected by each Raspberry Pi. In case one of the Pis, for instance, cannot detect the device due to it being out of range, the trilateration equation will be incomplete and not provide accurate data. Additionally, BLE devices such as Raspberry Pis usually have their receiver sensitivity lowered by design in order to save power. These two factors together mean that the effective sphere of detection for our setup is relatively small in an outdoor environment.

6.6 Ethical Considerations

When conducting experiments that involve collecting sensitive data, it is crucial to consider the ethical implications of the study. In this case, the experiments involved the collection of Bluetooth data, which can potentially reveal information about an individual's location and movements. To ensure that the data was collected in an ethical manner, the experiments were conducted in a relatively less crowded area, and after working hours when fewer people were outdoors. This was done to limit the number of people whose data would be recorded and to minimize any potential privacy violations.

Recording Bluetooth data in an outdoor environment is generally not considered a privacy violation, as the Bluetooth packets are transmitted as a broadcast. However, to further safeguard the privacy of individuals, efforts were made to limit the storage of personal data. If long-term storage of the data is necessary, it will be anonymised by obfuscating the MAC addresses and any other form of person data.

In addition to taking steps to limit the collection of personal data, precautions were taken when storing the data that was collected. The data was stored on a secure device with biometric authentication enabled. Backups of the data were made to cloud storage solutions with 2 Factor Authentication enabled. By taking these steps, it was ensured that the data was protected and could not be accessed by unauthorized individuals.

Finally, no form of analysis was performed on the other devices that were detected during the experimentation. Only the victim device was used for the analysis. This was done to further protect the privacy of individuals whose devices were detected but not relevant to the study. By focusing only on the data that was necessary for the study, it was ensured that the study was conducted in an ethical manner.

Chapter 7

Conclusion

This chapter concludes and summarizes the work.

7.1 Summary

In this project, we have contextually proven that it is possible to scan for and track unknown victims in an outdoor environment using an IoT device on their person. For this purpose, we built a system that scans Bluetooth devices on the 2.4GHz radio frequency range and reads packets of broadcast data from all available Bluetooth devices. Then we filter the data and use it to estimate and triangulate the precise location of a user in an outdoor environment. We explored the use of an extremely effective tool called "bettercap" that can perform Bluetooth spoofing, hacking and availability attacks. Moreover, all the devices and equipment were relatively cheap and inexpensive to procure and were readily available in the open market. Some programs were written in order to automate the task of filtering, analysing and deducing the position of the victim. These programs were also relatively inexpensive to run in terms of computing power required. In order to achieve this, we used standard reusable libraries as much as possible and standard mathematical estimation models wherever required and proved the hypothesis with relative accuracy. We have found a clear reflection in the data between the estimated positions using our developed system and the actual position of the victim throughout a range of distances from the scanners and in several different locations.

Chapter 8

Appendix A: Important Code Snippets

```
const fs = require('fs');
const csv = require('csv-parser');
const spline = require('spline-js');

// read the CSV file and parse it into an array of objects
const data = [];
fs.createReadStream('data.csv')
  .pipe(csv())
  .on('data', (row) => {
    data.push(row);
  })
  .on('end', () => {
    // extract the two columns as separate arrays
    const x = data.map((d) => parseFloat(d.column1));
    const y = data.map((d) => parseFloat(d.column2));

    // create the spline function
    const splineFunction = spline(x, y);
  });
```

Code for reading the calibration data and then building a spline function

```
function calculateXY(d1, d2, d3) {
  const x = [];
  const y = [];

  for (let i = 0; i < a.length; i++) {
    x.push((Math.pow(d1, 2)-Math.pow(d2, 2)+56.25)/15);
    y.push((Math.pow(d1, 2)-Math.pow(d3, 2)+56.25)/15);
  }
  return { x, y };
}
```

Code for calculating x and y coordinates given d_1, d_2, and d_3

```
async function main() {
    const start = new Date();
    const client = axios.create();

    const httpsAgent = new https.Agent({
        rejectUnauthorized: false,
    });

    const writer = createArrayCsvWriter({
        path: 'data.csv',
        header: ['MAC', 'Vendor', 'RSSI', 'Last Seen', 'Name', 'Alias', 'Cennectable', 'Flags', 'Services'],
        append: false,
    });

    await async.whilst(
        async () => (new Date()).valueOf() - start.valueOf() <= 34 * 1000,
        async () => {
            console.log(moment().format('YYYY-MM-DD HH:mm:ss'));

            try {
                const response = await client.get('http://127.0.0.1:8081/api/events', {
                    params: {
                        from: '',
                        n: '25',
                    },
                    headers: {
                        Authorization: 'Basic dXNlcjpwYXNz',
                    }
                });

                await async.eachSeries(response.data, async (row) => {
                    if (row.tag !== 'ble.device.new') {
                        return;
                    }
                    await writer.writeRecords([
                        [
                            row.data.mac,
                            row.data.vendor,
                            row.data.rssi,
                            moment(row.data.last_seen).format('YYYY-MM-DD HH:mm:ss'),
                            row.connectable ? 'TRUE' : 'FALSE',
                            row.data.flags,
                            row.data.services.join(','),
                        ],
                    ]);
                });

                await sleep(1000);
            } catch (e) {
                console.error(e);
            }
        }
    );
}
```

Code for extracting and saving BLE scanned data

Chapter 9

Appendix B: Raw Data From Raspberry Pi

MAC	Vendor	RSSI	Last Seen	Name	Alias	Cennectable	Flags
F3:E4:D1:29:BC:67	Anhui Huami Information Technology C	-67	2023-03-14 18:31:51	FALSE	BR/EDR Not Supported		
F7:28:0A:C3:52:47	Apple, Inc.	-52	2023-03-14 18:31:50	FALSE			
CE:45:00:11:7E:0E	Apple, Inc.	-89	2023-03-14 18:31:49	FALSE			
D6:EB:3D:BF:E0:76	Apple, Inc.	-74	2023-03-14 18:31:49	FALSE			
E0:54:58:F5:41:4B	Apple, Inc.	-71	2023-03-14 18:31:47	FALSE			
63:A8:73:6D:26:98		-89	2023-03-14 18:31:42	FALSE			
73:9B:BB:9F:D5:F9		-84	2023-03-14 18:31:47	FALSE			
65:07:6B:BC:09:A4		-85	2023-03-14 18:31:49	FALSE			
DD:34:8F:E3:94:9B	Apple, Inc.	-89	2023-03-14 18:31:42	FALSE			
6E:9A:9B:46:09:31		-89	2023-03-14 18:31:42	FALSE			
15:FD:02:C5:82:FB	Apple, Inc.	-80	2023-03-14 18:31:49	FALSE			
70:BA:12:1F:78:37		-84	2023-03-14 18:31:50	FALSE			
76:7B:B3:D6:9E:31		-89	2023-03-14 18:31:30	FALSE			
7E:E3:1F:ED:FC:67		-84	2023-03-14 18:31:49	FALSE			
F7:80:C8:9B:F7:4E	Apple, Inc.	-69	2023-03-14 18:31:48	FALSE			
73:A5:D2:2B:72:7D		-91	2023-03-14 18:31:44	FALSE			
FC:48:85:6D:0F:1C	Apple, Inc.	-83	2023-03-14 18:31:48	FALSE			
DA:C0:02:30:2D:7D	Apple, Inc.	-81	2023-03-14 18:31:50	FALSE			
08:42:A5:21:BD:8D	Apple, Inc.	-79	2023-03-14 18:31:51	FALSE	LE + BR/EDR (controller), LE + BR/EDR (host)		
54:CF:BA:26:66:E1	Apple, Inc.	-77	2023-03-14 18:31:51	FALSE	LE + BR/EDR (controller), LE + BR/EDR (host)		
68:3D:AD:28:74:D5	Apple, Inc.	-88	2023-03-14 18:31:45	FALSE	LE + BR/EDR (controller), LE + BR/EDR (host)		
FE:FA:DB:67:45:F8	Apple, Inc.	-92	2023-03-14 18:31:42	FALSE			
D7:1D:D1:DB:8F:56	Apple, Inc.	-90	2023-03-14 18:31:42	FALSE			
47:90:91:4C:57:00		-91	2023-03-14 18:31:48	FALSE			
08:12:10:17:7B:8B	Microsoft	-82	2023-03-14 18:31:51	FALSE			
F3:E4:D1:29:BC:67	Anhui Huami Information Technology C	-67	2023-03-14 18:31:51	FALSE	BR/EDR Not Supported		
F7:28:0A:C3:52:47	Apple, Inc.	-52	2023-03-14 18:31:50	FALSE			
CE:45:00:11:7E:0E	Apple, Inc.	-89	2023-03-14 18:31:49	FALSE			
D6:EB:3D:BF:E0:76	Apple, Inc.	-74	2023-03-14 18:31:49	FALSE			
E0:54:58:F5:41:4B	Apple, Inc.	-76	2023-03-14 18:31:51	FALSE			
63:A8:73:6D:26:98		-89	2023-03-14 18:31:52	FALSE			
73:9B:BB:9F:D5:F9		-84	2023-03-14 18:31:47	FALSE			
65:07:6B:BC:09:A4		-86	2023-03-14 18:31:52	FALSE			
DD:34:8F:E3:94:9B	Apple, Inc.	-89	2023-03-14 18:31:42	FALSE			
6E:9A:9B:46:09:31		-89	2023-03-14 18:31:52	FALSE			
15:FD:02:C5:82:FB	Apple, Inc.	-80	2023-03-14 18:31:49	FALSE			
70:BA:12:1F:78:37		-85	2023-03-14 18:31:52	FALSE			
76:7B:B3:D6:9E:31		-89	2023-03-14 18:31:30	FALSE			
7E:E3:1F:ED:FC:67		-84	2023-03-14 18:31:49	FALSE			
F7:80:C8:9B:F7:4E	Apple, Inc.	-69	2023-03-14 18:31:48	FALSE			
73:A5:D2:2B:72:7D		-91	2023-03-14 18:31:44	FALSE			
FC:48:85:6D:0F:1C	Apple, Inc.	-87	2023-03-14 18:31:52	FALSE			
DA:C0:02:30:2D:7D	Apple, Inc.	-71	2023-03-14 18:31:52	FALSE			
08:42:A5:21:BD:8D	Apple, Inc.	-87	2023-03-14 18:31:52	FALSE	LE + BR/EDR (controller), LE + BR/EDR (host)		
54:CF:BA:26:66:E1	Apple, Inc.	-85	2023-03-14 18:31:52	FALSE	LE + BR/EDR (controller), LE + BR/EDR (host)		

Raw data from Raspberry Pis

Bibliography

[1] Omar Alrawi et al. 'Sok: Security evaluation of home-based iot deployments'. In: *2019 IEEE symposium on security and privacy (sp)*. IEEE. 2019, pp. 1362–1380.

[2] *Arduino Due*. https://store.arduino.cc/products/arduino-due/. Accessed: 01-05-2022.

[3] Lu Bai et al. 'A low cost indoor positioning system using bluetooth low energy'. In: *Ieee Access* 8 (2020), pp. 136858–136871.

[4] Johannes K Becker, David Li and David Starobinski. 'Tracking Anonymized Bluetooth Devices.' In: *Proc. Priv. Enhancing Technol.* 2019.3 (2019), pp. 50–65.

[5] Xavier JA Bellekens et al. 'A Study on Situational Awareness Security and Privacy of Wearable Health Monitoring Devices.' In: *Int. J. Cyber Situational Aware.* 1.1 (2016), pp. 74–96.

[6] *Bettercap*. https://www.bettercap.org//. Accessed: 01-05-2022.

[7] *Bluetooth LE*. https://www.bluetooth.com/learn-about-bluetooth/tech-overview/. Accessed: 05-05-2022.

[8] Jingang Cao. 'A localization algorithm based on particle swarm optimization and quasi-newton algorithm for wireless sensor networks'. In: *Journal of Communication and Computer* 12 (2015), pp. 85–90.

[9] Ka Wai Cheung et al. 'Least squares algorithms for time-of-arrival-based mobile location'. In: *IEEE transactions on signal processing* 52.4 (2004), pp. 1121–1130.

[10] Ke Wan Ching and Manmeet Mahinderjit Singh. 'Wearable technology devices security and privacy vulnerability analysis'. In: *International Journal of Network Security & Its Applications* 8.3 (2016), pp. 19–30.

[11] *Data protection by design and by default*. https://gdpr-info.eu/art-25-gdpr/. Accessed: 05-05-2022.

[12] *Direction Finding*. https://www.bluetooth.com/learn-about-bluetooth/feature-enhancements/direction-finding/. Accessed: 05-05-2022.

[13] *ESP32-S3*. https://www.espressif.com/en/products/socs/esp32-s3/. Accessed: 01-05-2022.

[14] Ramsey Faragher and Robert Harle. 'Location fingerprinting with bluetooth low energy beacons'. In: *IEEE journal on Selected Areas in Communications* 33.11 (2015), pp. 2418–2428.

[15] Lothar Fritsch and Nils Gruschka. 'Extraction and Accumulation of Identity Attributes from the Internet of Things'. In: *Open Identity Summit 2021* (2021).

[16] Vincent Gao. *Bluetooth Proximity and RSSI*. https://www.bluetooth.com/blog/proximity-and-rssi. Accessed: 05-05-2022.

[17] Albert S Huang and Larry Rudolph. *Bluetooth essentials for programmers*. Cambridge University Press, 2007.

[18] Minhaj Ahmad Khan and Khaled Salah. 'IoT security: Review, blockchain solutions, and open challenges'. In: *Future generation computer systems* 82 (2018), pp. 395–411.

[19] Hyun-Jin Kim et al. 'A study on device security in IoT convergence'. In: *2016 International Conference on Industrial Engineering, Management Science and Application (ICIMSA)*. IEEE. 2016, pp. 1–4.

[20] Chenhui Li et al. 'An Indoor Positioning and Tracking Algorithm Based on Angle-of-Arrival Using a Dual-Channel Array Antenna'. In: *Remote Sensing* 13.21 (2021). ISSN: 2072-4292. DOI: 10.3390/rs13214301. URL: https://www.mdpi.com/2072-4292/13/21/4301.

[21] Guoquan Li et al. 'Indoor positioning algorithm based on the improved RSSI distance model'. In: *Sensors* 18.9 (2018), p. 2820.

[22] Ran Liu et al. 'Indoor positioning using similarity-based sequence and dead reckoning without training'. In: *2017 IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*. IEEE. 2017, pp. 1–5.

[23] Franco Loi et al. 'Systematically evaluating security and privacy for consumer IoT devices'. In: *Proceedings of the 2017 Workshop on Internet of Things Security and Privacy*. 2017, pp. 1–6.

[24] Sukhvir Notra et al. 'An experimental study of security and privacy risks with emerging household appliances'. In: *2014 IEEE conference on communications and network security*. IEEE. 2014, pp. 79–84.

[25] *Raspberry Pi Model 3B+*. https://www.raspberrypi.com/products/raspberry-pi-3-model-b-plus//. Accessed: 01-05-2022.

[26] *Raspberry Pi OS*. https://www.raspberrypi.com/software/. Accessed: 05-05-2022.

[27] Verónica Valeros. 'How bluetooth may jeopardize your privacy. An analysis of people behavioral patterns in the street.' In: (2013).

[28] Mathias Versichele et al. 'Pattern mining in tourist attraction visits through association rule learning on Bluetooth tracking data: A case study of Ghent, Belgium'. In: *Tourism Management* 44 (2014), pp. 67–81.

[29] Yapeng Wang et al. 'Bluetooth positioning using RSSI and triangulation methods'. In: *2013 IEEE 10th Consumer Communications and Networking Conference (CCNC)*. IEEE. 2013, pp. 837–842.

[30] Zheng Zuo et al. 'Indoor positioning based on Bluetooth low-energy beacons adopting graph optimization'. In: *Sensors* 18.11 (2018), p. 3736.