TOWARDS MORE SECURE BLUETOOTH LOW ENERGY BEACON TECHNOLOGY

by

EHAB ASHARY

B.S, King AbdulAziz University, 1998
M.S., University of Colorado Colorado Springs, 2013

A dissertation submitted to the Graduate Faculty of the University of Colorado Colorado Springs in partial fulfillment of the requirements for the degree of Doctor of Philosophy Department of Computer Science 2020
This dissertation for the Doctor of Philosophy degree by

Ehab Ashary

has been approved for the

Department of Computer Science

by

C. Edward Chow, Chair

Sang-Yoon Chang

Yanyan Zhuang

Jonathan Ventura

Francisco Torres-Reyes

Date: 2/12/2020
Towards More Secure Bluetooth Low Energy Technology

Dissertation directed by Professor C. Edward Chow

ABSTRACT

In this dissertation we investigated the security of Bluetooth Low Energy (BLE) technology, in particular the beacon systems used to identify geolocation of the users. The vulnerabilities of the current BLE beacon systems were analyzed and an attack application for exploiting such a vulnerability was developed. Three machine learning techniques are developed for identifying the signals of existing BLE beacons and distinguishing them from the those from attackers. To improve the identification, three filtering techniques are analyzed for their performance when working with those machine learning techniques. A new BLE protocol with PKI authentication features for Bluetooth 5.0 was proposed and developed for defending the BLE beacons systems and BLE system in general. Its performance was evaluated on a prototype system and was acceptable for real-time applications. Integrating the new BLE protocol and BLE Beacon Defense techniques into the secure data delivery to right location system was also discussed.

Keywords - Bluetooth Low Energy Beacons; Indoor Positioning Systems; Cybersecurity Attacks and Defences; Security; IoT; Encryption; Spoofing; Machine Learning
DEDICATION

To my parents, my sister, and my small family for their support throughout this tremendous journey.
ACKNOWLEDGEMENTS

First, I would like to thank my advisor and mentor, Dr. C. Edward Chow for his valuable guidance and direction. This thesis would not have been possible without his big support, kindness, and patience.

I would like to thank my dissertation committee member, Dr. Francisco Torres-Reyes, Dr. Jonathan Ventura, Dr. Yanyan Zhuang, and Dr. Sang-Yoon Chang for their constructive feedbacks and support.

Finally, big thanks go to my home university, King Abdelaziz University, Jeddah, Saudi Arabia that granted me a scholarship to get this degree, which I’m so grateful for.
# TABLE OF CONTENTS

**CHAPTER**

**CHAPTER 1 INTRODUCTION** .......................................................... 1

1.1 Overview .................................................................................. 1

1.2 Thesis Contributions ................................................................. 5

**CHAPTER 2 LITERATURE REVIEW** .................................................. 7

2.1 Indoor Positioning Using Received Signal Strength ....................... 7

2.2 Bluetooth Low Energy Beacon Security ..................................... 11

**CHAPTER 3 IMPORTANT BACKGROUND KNOWLEDGE** ............ 16

3.1 Indoor Positioning Technologies ................................................ 16

3.2 Measuring Principles and Positioning Techniques ....................... 17

3.3 Bluetooth ............................................................................. 20

**CHAPTER 4 ATTACKS ON BLE BEACON BASED INDOOR POSITION**

4.1 Introduction ............................................................................. 39

4.2 Contribution .......................................................................... 40

4.3 Attack Threats Against BLE Beacons ....................................... 40

4.4 Case Study ........................................................................... 44

4.5 Conclusion ........................................................................... 48

**CHAPTER 5 SELECTING FILTERS FOR BLE BEACON ATTACK DETECTION**

5.1 Introduction ........................................................................... 50
5.2 RSSI Analysis ................................................................................................... 50
5.3 RSSI Filters ....................................................................................................... 52
5.4 Filters Evaluations ........................................................................................... 58
5.5 Normal Operation Profiling Based on Moving Median Filter Evaluation ...... 67
5.6 Conclusion ........................................................................................................ 69

CHAPTER 6 BLE BEACON DETECTION ............................................................. 70

6.1 Introduction ....................................................................................................... 70
6.2 Architecture overview ..................................................................................... 70
6.3 Description of the BLE Tested Attacks ........................................................... 77
6.4 Beacon Swap Attack Detecting Method ......................................................... 78
6.5 Beacon Packet Injection Attack Detecting Method ........................................... 81
6.6 Beacon Moving Physical Attack Detecting Method ......................................... 85
6.7 Beacon Hide and Seek Attack Detecting Method ............................................ 87
6.8 Monitor Device Position Problem ................................................................... 88
6.9 Conclusion ....................................................................................................... 95

CHAPTER 7 ENHANCED BEACON PROTOCOL .............................................. 97

7.1 Introduction ..................................................................................................... 97
7.2 Proposed eBeacon Protocol ............................................................................ 98
7.3 Implementation and Experimental Setup ...................................................... 101
7.4 Security Analysis ............................................................................................ 103
7.5 Proposed eBeacon5 Protocol ........................................................................ 105
7.6 Conclusion ..................................................................................................... 110

CHAPTER 8 INTEGRATE E-BLE-BEACONS INTO R2D2 SYSTEM ................. 111
LIST OF FIGURES

FIGURE

Figure 1.1 Real-life use cases of BLE beacons ............................................................ 4

Figure 3.1 Bluetooth Classic, Smart Ready, and Smart adopted from [Permartne 2015] ........................................................................................................................................... 21

Figure 3.2 BLE channel map and WiFi adopted from [FUTURE CONNECTIVITY SOLUTIONS, 2014] .......................................................................................................... 22

Figure 3.3 How iBeacons Work adopted from [estimote.com] ................................. 25

Figure 3.4 iBeacon advertisement packet adopted from [ARMmbed 2015] .......... 26

Figure 3.5 Four Ranging Proximity States adopted from [Koning 2014] ............... 30

Figure 3.6 Legacy Advertising and Advertising Extensions ...................................... 36

Figure 3.7 Advertising packet chaining ..................................................................... 36

Figure 4.1 Output of the scanning program ............................................................... 43

Figure 4.2 Shopping center application error message .............................................. 45

Figure 4.3 The output of the scanning program ........................................................ 46

Figure 4.4 Broadcasting cloned iBeacon ID .............................................................. 46

Figure 4.5 The shopping mall application after cloning ........................................... 47

Figure 4.6 ................................................................................................................... 48

Figure 5.1 (a): RSSI Variation ................................................................................... 51

Figure 5.2 The ongoing discrete Kalman filter cycle. The time update projects the current state estimate ahead in time. The measurement update adjusts the projected estimate by an actual measurement at that time ................................................................. 54

Figure 5.3 Floorplan of experiment area and beacon locations ............................ 59
Figure 5.4 Impact of the Filtering Processes .............................................................. 60
Figure 5.5 Raw Date and SVM .................................................................................. 64
Figure 5.6 Raw Data and Random Forest ................................................................. 64
Figure 5.7 Raw Date and Decision Tree ................................................................. 64
Figure 5.8 Moving Median and SVM ........................................................................ 64
Figure 5.9 Moving Median and Random Forest ........................................................ 65
Figure 5.10 Moving Median and Decision Tree ........................................................ 65
Figure 5.11 Moving Average and SVM ..................................................................... 65
Figure 5.12 Moving Average and Random Forest ..................................................... 65
Figure 5.13 Moving Average and Decision Tree ....................................................... 66
Figure 5.14 Kalman Filter and SVM .......................................................................... 66
Figure 5.15 Kalman Filter and Random Forest .......................................................... 66
Figure 5.16 Kalman Filter and Decision Tree ............................................................ 66
Figure 6.1 Architecture of the proposed solution ....................................................... 71
Figure 6.2 Overview of the proposed beacon swap attack detection method ............ 79
Figure 6.3 Beacon Swapping attack testbed ............................................................... 80
Figure 6.4 Beacon packet injection attack testbed ..................................................... 83
Figure 6.5 Testbed room with 4 beacon installed ...................................................... 89
Figure 6.6 Testbed in 2-dimensional space ............................................................... 90
Figure 6.7 Beacon Clique Relation for Point 12 .......................................................... 92
Figure 6.8 Best solution recommended by our algorithm ......................................... 94
Figure 6.9 The final position of a monitoring device ................................................ 94
Figure 7.1 eBeacon packet ......................................................................................... 99
Figure 7.2 eBeacon operation.....................................................................................100
Figure 7.3 eBeacon5 packet .....................................................................................108
Figure 8.1 Right Place Right Path Data Delivery. ...................................................113
Figure 8.2 The building with wifi base stations (greendots).................................115
Figure 8.3 RSSI values collected from the mobile client and Aruba system..........116
Figure 8.4 Proximity Positioning Technique. ..........................................................118
Figure 8.5 BLE Positioning Method. .......................................................................119
Figure 8.6 BLE Room Layout ..................................................................................120
Figure 8.7 The Message Flow Between the Entities of the Proposed Solution. ......121
Figure 8.8 Generic Location Verification System.....................................................123
Figure 8.9 An overview of OTMEN architectural components............................125
Figure 10.1 Angel of Arrival (AoA) and Angel of Departure (AoD) [Bluetooth Core Specifiation v5.1] .................................................................130
LIST OF TABLES

Table 3.1 UUID, Major, and Minor example.............................................................28
Table 3.2 iBeacon vs Eddystone: Comparison.......................................................33
Table 3.3 Different advertising PDUs......................................................................38
Table 5.1 F1 Scores for Classification Algorithms with Different Filtering Methods .................................................................................................................................63
Table 5.2 Evaluation Metrics for Classification Algorithms with Different Window Size ........................................................................................................................................68
Table 6.1 Values of Path Loss Exponent for different environments...............75
Table 6.2 The results of swapping attack detection method ................................81
Table 6.3 The actual and calculated distance between the monitor device and cloned beacon ...........................................................................................................................................84
Table 6.4 Moving Attack detection results ............................................................87
Chapter 1

Introduction

1.1 Overview

In recent years, Location Based Services (LBS) have been widely applied in different mobile applications and used by many to enrich their social lives, navigations, finding restaurants, checking public transportation and other usages.

Global Positioning System (GPS) [B.Hofmann1997] is the most widely used satellite based positioning system and supported by different devices and mobile phones. GPS can provide accurate positioning estimations for most of the outdoor positioning and applications as long as line of sight transmission between receivers and satellite is possible. Consequently, GPS generally only works well in outdoor environments and cannot be deployed for indoor use [P.Zandbergen2009]. Thus, Indoor Positioning System (IPS) is required to support location-based services when users or objects are located in indoor areas.

Indoor Positioning System (IPS) is referred to a navigation system, which is made of network devices to locate objects or people inside indoor environment [S.Ingram2006]. In addition, Dempsey [M Dempsey, 2003] defines an IPS as a system that continuously and in real-time can determine the position of something or someone in a physical space such as universities, hospitals, gymnasium, etc. In other words, IPS should work all the time unless the user turns off the system and offer updated position information of the target and estimate the positions within a minimum time delay. So, someone might say basically IPS is another positioning system and implementing such a system wouldn’t be a difficult task since it is already have been done outdoor. Actually, that is not true.
In comparison with outdoor environments, indoor environments seem more complex because of the building geometry and the presence of various obstacles, such as walls, furniture, and human, could influence the propagation of the signal and lead to multi-path and delay problems [Ladd2004]. Also, due the existence of different object, indoor environment relays on non-line-of-sight (NLOS) propagation in which signals cannot travel directly in straight path from the sender to the receiver, which could cause inconsistent time delays at the receiver. Furthermore, the signal strength tends to fluctuate due to the existence of various interference sources such as wireless devices, mobile devices, fluorescent lights, and microwave ovens [Wang, 2017]. Considering these issues, IPSs raise new challenges and more questions to be answered.

There are diversity of different technological solutions for indoor positioning system, which includes camera [Mulloni, 2009], Infrared [Hauschildt, D. and Kirchhof, N. 2010], ultrasound [Sato et al. 2011], WIFI [Bahl and Padmanabhan 2000], Radio Frequency IDentification (RFID)[Peng et al.2011], Ultra Wideband (UWB) [Wang et al. 2010], ZigBee [Larrañaga et al. 2010], and Bluetooth [Bargh and de Grote 2008].

Despite the availability of different technologies, we are still away from achieving a cheap global indoor positioning solution based on a single technology, such as that provided by GPS, and deliver accuracy of 1 meter or less [Mautz, 2012]. Thus, there is a need for other technology to be tested, and there are still opportunities for further enhancements.

Bluetooth Low Energy (BLE) beacon technology, that repeatedly sending static advertisement packets, is one of the new technologies that can be used to develop indoor positioning systems. BLE capable devices are nearly already global as all smart phones
and PCs from major manufacturers have adopted Bluetooth 4.0, the standards that support BLE [Ensw 2015]. In addition, Abi research report titled “BLE tags and location of things” outlines that by 2019 BLE beacon will create a 60-million-unit market [Gallen. 2014]. As a successor to its previous version, Bluetooth Classic where its primary aim is to provide effective high data rate for audio and data streaming, BLE protocol has been developed by the Bluetooth SIG to support energy efficient and low data rate devices which makes it suitable for power constrained Internet of thigs (IoT) applications and devices. Since BLE unifies the advantage of both unmanned power constrained IoT application and Bluetooth enabled smart devices, it has gained much research interests from both academia and industrial and many interesting applications of BLE beacon have been proposed and developed, namely, improving the users shopping experience [Zaim2016] guiding in museums [Alle2016], Indoor positioning for warehouse management [ZHAO2016], helping blind [CHER2017], and keeping tracking of senior citizens’ activities information [KASH2017]. Furthermore, increasing number of BLE infrastructures are being deployed for commercial uses; some of these real-life deployment examples are shown in Figure 1.1.
Figure 1.1 Real-life use cases of BLE beacons

(a) Indoor navigation system with augmented reality in Gatwick Airport, England. (b) Buses in Luxembourg leverage beacons to push bus schedules. (c) Shopping mall Lyngby Storecentre indoor treasure hunts app. (d) University of Texas beacon app. (e) Indoor navigation app for Sarasota Memorial Hospital, Florida.

In addition, Proximity. Directory’s Q1 2017 Report [Prox2017] noted that BLE beacons are the most popular technology in the industry. With 14,486,00 sensors are registered to have been deployed globally, beacon technology accounts for 65% of the total, with WIFI points at 20% and 15% for NFC.

It is clear that beacons based on BLE standard have recently been gaining acceptance as preferred enablers for proximity sensitive experiences. However, one of their major security weaknesses is broadcasting a static beacon ID, which in turn can lead to a range of attacks.
Spoofing is one of BLE beacon vulnerabilities. Since a beacon broadcasts its static ID publicly, any third-party could capture a specific beacon’s ID and impersonate that beacon anywhere they desire.

1.2 Thesis Contributions

Although there is a potential growth in adopting BLE beacon technology for application related to location information. Unfraternally, it appears that the vast majority of the existing beacon protocols miss features to protect the transmitted beacon messages. In addition, limited studies have highlighted the potential security threats and proposed countermeasures. Therefore, methods of detecting attacks on BLE infrastructure must be developed. A security protocol is desired to provide protection to BLE beacons.

In this thesis, we address the following question:

*Can we use trusted monitoring device in order to detect spoofing and physical attacks on the BLE infrastructure by implementing and utilizing machine learning algorithms? In addition, Can the current BLE beacon protocol to be modified to authenticate the broadcasted beacon messages?*

The effort for answering the question leads to the following N contributions:

**Contribution 1:** Provide a survey of existing BLE beacon Technology and an analysis of their vulnerabilities.

**Contribution 2:** Present tools and application that exploit such a BLE beacon system vulnerability.
**Contribution 3:** Analyze the effectiveness of filtering techniques to work with machine learning algorithms for the identification of BLE Beacons.

**Contribution 4:** Present a defense mechanism for detecting the attacks on BLE beacon systems.

**Contribution 5:** Present the design and development of an authentication protocol for Bluetooth 4.0 and 5.0 systems which is capable of detecting the fake BLE beacons.

**Contribution 6:** Discuss how the BLE beacon system can be used to enhance the secure data delivery to right locations applications.

This thesis is organized as follow: Chapter 2 discusses the background knowledge and information. Chapter 3 presents the related work. Chapter 4 describes a mobile attack application that can mislead a BLE beacon-based geolocation system. Chapter 5 analyzes the effectiveness of filtering techniques to work with machine learning algorithms. Chapter 6 presents a defense mechanism for detecting the attacks on BLE beacon systems. Chapter 7 discusses the design and development of an authentication protocol for Bluetooth systems. Chapter 8 presents the lessons learnt. Chapter 9 contains the future directions. Chapter 10 is the conclusion.
Chapter 2

Literature Review

In this chapter, we provide background on the following topics: Indoor positioning using received signal strength, and Bluetooth low energy security

2.1 Indoor Positioning Using Received Signal Strength

In this thesis we mainly consider the receiver signal strength (RSS) indicators in order to estimate the distance between an object and a relative point. Two of the positioning techniques that could utilize RSS for location measurement are (1) fingerprinting positioning technique and (2) alteration, in which the signal between a transmitter and receiver devices attenuates logarithmically in terms of the distance between them and the signal strength can be described as a function of distance [Hata1980], [Nurminen2012]. In addition, two of the positioning technologies use RSS for location measurement are WLAN and Bluetooth. However, since we will use Bluetooth Low Energy (BLE) for our work, this related work section focuses on comparing between using WLAN and BLE technologies for indoor positioning. In addition, the challenges of using RSS for distance measurement.

2.1.1 WALAN Technology using RSS for indoor positioning

RADAR indoor localization system is a pioneer work in WLAN fingerprinting [Bahl and Padmanabhan2000], which uses the signal strength and K-Nearest Neighbor (KNN) fingerprinting location algorithm. The system can provide 2D absolute position
information with accuracy about 4m with about 50% probability. Their findings include that the performance depends on the data points taken and the orientation and speed of the user. In addition, there is no consideration of privacy issues in the design of RADAR system, where a person using a device with WLAN interface may be tracked even (s)he doesn’t want anyone to know her/his location.

In 2006 King et al. proposed The COMPASS system [King et al 2006]. Another indoor WLAN fingerprinting system that uses RSS to locate a user carrying a WLAN enabled device. The main contribution of the COMPASS system is that the user’s orientation and the blocking effects from the user are considered in the location sensing process by using a digital compass. With extensive pre-calibration, 1 m grid of calibration points, 110 RSSI measurements at each calibration point in 8 different orientations, the system achieved 1.65 m accuracy. In addition, the system only considers tracking a single user. In other words, the location of multiple users at the same time has not been discussed.

In 2010 Gansemer et al. developed an adapted Euclidean distance model, which takes into consideration a dynamically changing environment with changing sets of base stations. Therefore, the distances in the model are normalized according to the number of the stations received. Their proposed fingerprinting methods can provide location estimation error lies at 2.12m and the calibration effort may be decreased by a factor of 4 if an increase of location estimation error from 2.12m to 2.69 is tolerated.

The effectiveness of WLAN based fingerprint system deployment is characterized mainly by the required time and efforts for site survey and maintenance of the fingerprint database to keep it up to date. In addition, the accuracy or WLAN RSS based positing
system can be affected by the attenuation of WLAN signal by heterogamous devices, orientation and people.

According to He and Chan the major factors that affect the RSS measurement are WLAN chipset sensitivity and antenna installation position [He and Chan 2016]. Viel and Asplund observed differences in RSS of up to 12 dBm from same signal measured by different smartphones [Viel and Asplund, 2014] Therefore, in 2013 Hossain et al presented new approach for RSS calibration on heterogeneous devices. The approach is based on new location fingerprint, signal strength difference (SSD), which is free from hardware specific parameters [Hossain et al 2013] the SSD fingerprinting is also applicable for BLE based positioning.

In order to study the effect of the device orientation, Chapre et al measured RSS on a stationary laptop from 13 AP in an office environment where the data was collected for 4 different directions during 24 hours at a fixed location. The results show that the RSS variation is on average 3dBm[Chapre et al2013].

In [Della Rosa2012] the authors studied the body-lose effect on RSS measurement on smartphones. The results show that the signal strength at a given location varies by up to 5dBm depending the user orientation because the user’s body creates a systematic source of error in RSS measurement. In addition, if the antenna is fully covered by the user, the RSS error can reach 12 dBm that corresponds to approximately 9m of error according to the authors.

2.1.2 BLE technology using RSS for indoor positioning

Indoor positioning based on Bluetooth Low Energy can use the same methods as WLAN based positioning. In 2015 Faragher and Harle explored the use of BLE beacons
for fingerprint positioning. They have achieved tracing accuracies of less than 2.6 m 95% of the time using a dense beacon distribution, 1 beacon per 30 m² and less than 4.8 m using 1 beacon per 100 m² in an office environment that covered approximately 600 m². On the other hand, WiFi achieved less than 8.5m 95% of the time for the same environment [Faragher and Harle, 2015].

Zhao et al [Zhao et al, 2014] conduct extensive experiments in indoor environments and compared the positioning accuracy of WiFi and BLE in almost identical environment and conditions. The results showed that BLE based positioning is more accurate than WiFi by around 27%. In addition, the authors conclude that the superiority of BLE in indoor localization over WiFi could be that BLE has 1) channel hopping mechanism, 2) lower transmission power, and 3) higher sampling rate than WiFi. Channel hopping mechanism can average out interference in a given channel or completely skip the channel interference when the BLE transceivers would hop to the next channel for communication. However, WiFi cannot skip the interference because it has no channel hopping mechanism. As a result, the RSS of BLE is less noisy and cleaner than the RSS of WiFi. In addition, the lower transmission power of BLE provides better performance of localization because it can reduce the multipath effect in some scenarios. Finally, the higher sampling rate makes it possible to average out the occasional outliers caused by interference or multipath effect, which improves the positioning accuracy. However, in order to smooth the measured RSS values, different smoothing techniques could be used. For example, weighted moving average [Liu et al2014], Kalman filter [Lee et al2016], and particle filter [Dickinson et al, 2016].
The emergence of Bluetooth Low Energy (BLE) beacons opens up a new generation of indoor positioning systems that can be more accurate, reliable and efficient. ABI Research’s report, ”BLE Tags: The Location of Things” states that total BLE beacon shipments could exceed 400 million units in 2020. The Bluetooth Special Interest Group predicts that by 2018 more than 90 percent of Bluetooth-enabled smartphones will support the BLE standard, so this technology will become ubiquitous. However, since BLE uses receiver signal strength for localization, building geometry, device orientation, signal fluctuation and presence of various obstacles such as doors, walls, people still considered as challenges and could affect the localization accuracy.

2.2 Bluetooth Low Energy Beacon Security

BLE beacons have come into the spotlight of both industrial and academic sectors. Therefore, studies have highlighted the potential security threats and limited countermeasure studies have yet been proposed.

In [Liu2016] The authors used Hidden Markov Model (HMM) to estimate the probability of a beacon device being physically removed, relocated and being cloned. With a false alarm rate of 5%, they have been able to detect removed/stolen beacons perfectly, and relocated or cloned beacons could be detected with around 70% accuracy. However, for the proposed architecture to be functional, the user’s presence near the infrastructure deployment site is necessary. Their architecture based on building a profile of the original beacon location sequence from different users randomly walk between two bookshelves in order to build training data. For detection, a user needs to walk again between the bookshelves and use an app to upload the beacons signal as he/she walks. The uploaded
profile forms a sequence of beacons that the user passed by. The detected sequence will be compared to the previously collected during training and HMM is used to determine if the detected sequence is unusual or not.

The problems in this design that as users walk around the covered area, they may not access the abnormal beacon in the entire sequence which could lead to longer detection. In other words, there could be a cloned beacon somewhere, but the system couldn’t find it because nobody reported it yet. In addition, in their experiment, they collected 93 sequence of randomly walking between two bookshelves to build their training data for 12 beacons installed but how many randomly walking sequences are needed to cover larger area. This lays doubt on the practicality of the proposed system as real beacon infrastructure. Finally, depending on untrustworthy users for building a security model is not a good practice. What if the user tempered the reported sequence before sending it to the server?

In [Misr2015] the authors demonstrated a packet injection attack on a BLE beacon based indoor location system for regular environment. The location heatmap illustrate a change in the distributions of occupations inside the area because of the attack. Similar work in [Olif2017], the authors evaluate the impact of spoofing and swapping attacks on their BLE based occupancy system. Their results show that both attacks have a negative impact on their classification accuracy.

Another work for securing BLE beacons is proposed [Jeon2018]. The authors proposed a crowed assisted architecture for dynamically modifying BLE beacon IDs as a countermeasure against beacons spoofing and piggybacking attacks. The proposed architecture is composed of three main components 1) beacon ID manager, 2) mobile SDK, and 3) synchronization and monitoring protocol. The beacon ID manager is an online
server that contains information regarding the beacon IDs. It is responsible for identifying which beacons that require modification and scheduling them for modification. The user’s mobile SDK will allow the user’s smartphone to communicate with the beacon ID manager to retrieve a list of the beacons that need to be modified and execute the required commands to update the beacons ID when the user’s mobile receive beacon signals. Finally, the synchronization and monitoring protocol is responsible after a successful beacon ID modification event to update the content mapping information on the beacon ID manager. The user’s mobile application monitors the beacons and transmit the IDs to the online server all the time. The online server reads those values and check for a successful modification event. Once a modification has been observed, the beacon ID manager data will be changed.

The limitation of this architecture that’s in order for the system to change the beacon ID, a user’s presence is required. Otherwise, the beacon will keep broadcasting the same ID and vulnerable to be cloned. An interesting attack against this architecture would be to clone one of the beacons and monitor the behavior of the system when the user tries to update the IDs for both beacons the original and the fake one.

Finally, A BLE beacon prototype that adds a secure signature to the advertisement packet is described in [Schu2016]. However, the length of the chosen signature is 64 bytes. Therefore, they had to break down the signature across three packets instead of using one advertising packet since the maximum payload that could be transmitted using BLE advertising packet is 31 bytes.

Different countermeasures to spoofing and piggybacking attacks have been proposed by some BLE beacon manufactures. In [Gilc2014] the author proposed a geolocation
validation approach in which the geolocational information of individual beacons are pre-
registered on an online server. On the user side, location information provided by GPS
module on the user mobile is transmitted to the sever along with detected beacons signals
in order to ensure the physical presence of the user near the detected beacon. However, the
GPS location can be inaccurate in indoor environment.

Another approach proposed by Kontakt.IO, one of the BLE beacon hardware
manufacturer, is to shuffle some of the fields in the advertisement packet. Similarly,
Estimote, another hardware manufacturer, has a solution which also rotates the UUID of
the beacon. Both solutions rely on secure access to their cloud services for beacon
identification. The technical details of these methods were not fully disclosed

Finally, in 2016 researchers from Google published a white paper introducing the
concept of cloud passed AES-encrypted eight-byte Ephemeral identifiers change pseudo
randomly over a range from $2^{10}$ to $2^{15}$ second in order to protect their beacon protocol
Eddystone [Hass2016]. The beacon has to be registered to Google Proximity Beacon API
resolving service that shares Ephemeral Identity Key with the individual beacon, so the
identifier can be resolved to useful ID. The identifier can be resolved to useful information
by a service that shares a key (the Ephemeral Identity Key, or EIK) with the individual
beacon. Any use of Eddystone-EID requires both a beacon and a resolving service (such
as the Google Proximity Beacon API). One of the major limitations of their concept. In
order to resolving the received EID the resolver needs to try all keys it has to compute its
own version of the EID and compare it with the received ID.

From the state of art and its limitation, we can conclude that there is still a need for
secure system that able detect potential threats and attack and respond appropriately while
being able to take some form of precautionary measures [Kim2015, Misr2015, Koli2017, Tay2016].
Chapter 3

Important Background Knowledge

In this chapter, we provide important background knowledge on the following topics: Indoor positioning technologies, measuring principles and positioning techniques, and various topics related to Bluetooth technologies.

3.1 Indoor Positioning Technologies

Different technologies can be used for indoor positioning, which includes camera Infrared, ultrasound, WIFI, (RFID, Ultra Wideband (UWB) ZigBee, and Bluetooth. However, researchers classified indoor positioning technologies in many different ways. In 2003 Collin et al., classified the indoor positioning technologies into two classes, technologies that require special hardware in the building and self-contained technologies [Collin et al 2003]. On the other hand, Gu et al, divided them into two classes according to their need for existence of networks: (1) network-based and (2) non-network based technologies [Gu et al, 2009]. The network based technologies take advantage of the existing network infrastructure, where no additional hardware infrastructure is needed, which could be preferred for cost reasons. However, the non-network based technologies use dedicated hardware for positioning and have the freedom of the physical specification by the designers, which may offer higher accuracy. The authors also classified the indoor positioning technologies according to the system architecture into three classes (1) Self-positioning architecture, in which objects calculate their positions by themselves and take advantage of the infrastructure, which would provide higher security and privacy for the
objects; (2) infrastructure positioning architecture, which estimate the location of the objects automatically using the infrastructure to find if the target is in the coverage areas without the need for the object to initiate the positioning request; (3) self-oriented infrastructure-assisted architecture, in which an object sends a request to the positioning system to start the positioning measurement and then gets its location information from the system. In other words, no positioning activates for the object can be carried out unless the object allows the positioning system to track it. In addition, they classified indoor positioning technologies into six classes according to the main medium: (1) infrared (IR) technologies; (2) ultra-sound technologies; (3) radio frequency (RF) technologies; (4) magnetic technologies; (5) vision-based technologies and (6) audible sound technologies.

In 2011 Chóliz et al., classified the indoor positioning technologies into two classes; (1) parametric where a location is computed based on prior knowledge and (2) non-parametric where a location is computed by processing the data taking into consideration some statistical parameters [Chóliz et al. 2011]. Instead, in 2016 Alarifi et al., classified the indoor positioning technologies into two main classes (1) building dependent, where the technologies depend either on an existing technology in the building or on the map and structure of the building; (2) building independent technologies that do not require any special hardware in a building. [Alarifi et al., 2016]. In order to employ these technologies, different positioning techniques and algorithms had been developed.

### 3.2 Measuring Principles and Positioning Techniques

In general a location system could provide four kinds of location information; (1) physical location, which is expressed in the form of coordinates that identify a point on a 2D/3D map; (2) symbolic location, which express a location in a natural language way,
such as in the coffee shop, in the second floor or at home; (3) absolute location, where a shared reference grid is used to allocate an object; (4) relative location information which is usually based on the proximity to known reference points or base stations [Hightower and Borriello, 2001].

In order to provide this information, there are four different techniques could be employed individually or in combination: triangulation, fingerprinting, proximity and vision analysis. Triangulation, fingerprinting, and vision analysis can provide absolute and relative location information. However, the proximity positioning technique can only offer relative information.

The triangulation positioning technique uses the geometric properties of triangles to compute object locations. This technique can be divisible into the subcategories of (1) lateration and (2) angulation. In alteration an object’s position can be computed by measuring its distance from multiple references points. Instead of measuring the distance directly using received signal strengths (RSS), time of arrival (TOA) or time difference of arrival (TDOA) is usually measured, and the distance is derived by computing the attenuation of the emitted signal strength or by multiplying the radio signal velocity and the travel time. Round trip time of flight (RTOF) or received signal phase method is also used for range estimation in some systems. In angulation in order to locate an object angel of arrival (AOA) is used to compute angels relative to multiple reference points. RSS, TOA, and TDOA need to know the position of at least three reference elements to perform location estimation. On the other hand, AOA only require two. However, AOA needs relatively large and complex hardware requirements in addition AOA is highly range
dependent. If the target object to be located is far away, the AOA technique may contain some errors that would result lower accuracy [Chen et al 2006].

The fingerprinting positioning technique uses pre-measured location related data. Fingerprinting includes two phases (1) offline training phase and (2) online position determination phase. In the offline phase, useful location related data with respect to different places in the position estimation area is measured and collected for the position estimation, for example WiFi access point RSS values can be measured at different location and saved in a database. During the online position determination phase, the location related data of a target object is measured and compared with the pre-measured data collected in the offline phase to get a similar case in the database to make the location estimations. The most common fingerprinting-based algorithms are Neural Networks, Support Vector Machines (SVM) and K-Nearest Neighbors (k-NN) [Zhang et al 2013].

The main drawback of fingerprinting systems is that changes in the environment or even people may force recalculation of the pre-measured data. In addition, the offline training phase is a time and manpower consuming.

The proximity positioning technique examines the location of an object when it nears a known location. The proximity location technique needs to fix a number of detectors at the known positions. When a detector detects a tracked object, the position of the object is considered to be in the proximity area marked by the detector. The provided proximity location information could accurately specify whether a targeted object is in the room or not. However, it is not enough to provide the absolute location.

The vision analysis estimates a location from the image received by one or multiple points. Usually, one or multiple cameras are fixed in the tracking area to cover the whole
place and take real-time images. From the images, the targeted objects are identified. The observed images of the objects are looked up in the pre-measured database to make the position estimations.

3.3 Bluetooth

Bluetooth is a wireless technology invented in 1994 operates in the unlicensed Industrial, Scientific and Medical (ISM) band at 2.4GHz, and was later standardized as IEEE 802.15.1. However, the Bluetooth Special Interest Group (SIG), which has more than 30,000 member companies in the areas of telecommunication, computing, networking, and consumer electronics, is overseeing the Bluetooth specifications.

Bluetooth is designed to operate at lower speed, short range and less power in order to establish a link and transfer data between devices. Bluetooth speed varies from below 1 Mbit/s to 100Mbit/s and range is between 0.5 to 100 meters. The speed and range the Bluetooth that could achieve depends on the transmission power and the Bluetooth version.

The most implementations of the specifications are Bluetooth Basic Rate/Enhanced Data Rate (BR/EDR), which is also known as Bluetooth Classic and was adopted as version 2.0/2.1, and Bluetooth with low energy (BLE), which is also known as Bluetooth Smart and was adopted as version 4.0/4.1/4.2 in 2010. Each implementation uses different chipsets to meet essential hardware requirements. In other words, the Bluetooth Classic chipsets don’t support the new 4.0 standard including the low-energy operations required. However, Dual-mode chipsets are also available for applications that include both use cases. Devices with dual-mode chipset are called Bluetooth Smart Ready. In other words, Smart Ready devices work with the classic devices, but they are also “ready” for the new ones as in Figure 3.1
Both Bluetooth Classic and BLE operate in the 2.4GHZ ISM band that extends from 2402MHz to 2480MHz. However, the number of the used channels is reduced from 79 channels of 1 MHZ each in Bluetooth Classic to 40 of 2 MHZ channels (PHY channels) in BLE. 37 of these channels are used for data exchange during a connection and the last three channels (37,38, and 39) are used as advertising channels to setup connections and send broadcast data.

During BLE advertisement, the BLE broadcast device transmits packets on the three advertising channels one after the other, which would help scanning devices to listen to those channels and discover nearby devices.

Channels 37, 38 and 39 are spread across the 2.4GHZ band on purpose. 37 and 39 are the first and the last channel in the band and 38 is in the middle. Therefore, if any of the advertising channels is blocked, the others are likely to be free since quite a few MHZ of bandwidth separates them. In addition, channel 38 is placed between WIFI channels 1 and 6 in ordered to minimize the WIFI signal interference as in Figure 3.2
Since the 2.4GHz band has become crowded, a frequency planning is needed in order to reduce interference and improve robustness. Therefore, BLE uses frequency hopping spread spectrum (FHSS) technique, in which the radio hops between channels on each connection using the following formula:

\[
\text{Channel} = (\text{current channel} + \text{hop}) \mod 37
\]

The value of the hop is communicated when the connection is established consequently it is different for every new established connection. (Townsend 2015)

Finally, BLE varies from Classic in the following:

- **Power consumption**: BLE power consumption is significantly reduced to 50-99% of Classic power consumption since BLE devices consume between 0.01 to 0.5W compared to 1 W in Bluetooth Classic.
- **Lower cost**: BLE chips are 60 to 80% cheaper than Bluetooth classic chips
- **Faster connection**: in BLE devices can connect and send and acknowledge data in 3ms. On the other hand, Bluetooth Classic needs more than 100 ms.
Applications: BLE can achieve less than 0.3Mbit/s data rates, which make it ideal for applications that require small periodic transfers of states. On the other hand, Bluetooth Classic can achieve between 0.7Mbit/s to 2.1Mbit/s data rate. Consequently, it is preferred for applications require consistent communication and more data throughout.

3.3.1 Beacon devices

Beacons are devices consist of microcontroller and BLE radio chips that use BLE in order to broadcast stored data at regular time intervals and usually operate on coin-cell batteries. However, other power options such as AA batteries or external power supply are available and can be used.

BLE devices could operate in four different device roles:

- A Peripheral: an advertiser that is connectable and could operate as a slave in a connection.
- A Central: a master device in a connection that scans for advertisements and can initiate connections.
- A Broadcaster: a non-connectable device that can broadcast data.
- An Observer: device that scans for advertisements but cannot initiate connections [Texas Instruments, 2016]

Beacons can act as Peripheral and Broadcaster. In both roles, the device sends the same type of advertisements with a flag to indicate whether it is connectable or non-connectable.

In non-connectable beacons, the device is in broadcasting mode transmitting stored information. Because in this mode the beacon doesn’t activate any receiving capabilities, it could achieve the lowest possible power consumption by simply waking up, transmit data and going back to sleep. On the other hand, connectable beacons are in peripheral
mode so they can transmit and receive. This mode allows a central device, for example a smartphone, to connect and configure the installed firmware.

Every beacon had a specific firmware, programed code that enables the hardware to operate and control several features that impact the battery life. These features include:

- **Transmitting Power (TX power)/Broadcasting Power**: the power that a beacon broadcast a signal and given in dBm. The value typically ranges between -40 dBm and +10dBm. High broadcasting power will improve the distance that the signal can travel but would negatively impact battery life. The distance that a beacon could cover range from a few centimeters to up to 50m depends on the radio chipset and surrounding environment.

- **Advertising Interval**: the rate that a beacon broadcast a signal, which typically ranges between 100ms to 2000 ms. An interval of 100ms means the signal is broadcasted every 100 milliseconds or 10 times in a single second. A higher interval of 2000ms means the signal is broadcasted every 2 seconds. Short advertising interval could improve precision and responsiveness but would negativity impact battery life.

### 3.3.2 BLE Protocols and Profiles

On top of the BLE protocol stack there are numbers of protocols exist, which specify the frames types and how beacons can communicate with receivers devices. Two of the dominant protocols are Apple’s iBeacon and Google’s Eddystone. Table 3.2 highlights some of the differences between the two protocols. However, the main difference is the number of the packet type broadcasted by the beacon. iBeacon needs one packet type to work. One the other hand, Eddystone supports four types (UID, URL, TLM, and EID)

#### 3.3.2.1 iBeacon

Apple announced iBeacon in December 2013 as the first BLE beacon protocol to come out, which provides proximity location service based on expected received signal strength indication (RSSI). It is a proprietary and closed standard supported in several Apple SDKs
and can be broadcasted and read by any BLE enabled device that runs iOS not older than iOS7 or Google android 4.3. Since iBeacon is a proprietary protocol, Apple provides the developers with a sample to use API to integrate the protocol into their applications.

iBeacon devices are transmitting only devices. They work by periodically transmitting an advertisement packet that contains the measured power and numerical identifiers, which are mapped into actions by an application on a mobile device. The typical scenario could be like this. A consumer carrying a smartphone walks into a store. On the consumer’s smartphone, an application listen for iBeacons. When the application discovers an iBeacon device, it sends the relevant data (the measured power and numerical identifiers of the iBeacon) to its server, which then locates the consumer and triggers an action based on the location. This could be something as simple as a push message [“Welcome to Macy’s”], and could include other things like targeted advertisements, special offers, and helpful reminders [“It is your anniversary. Don’t forget to buy a gift”] Figure 3.3. iBeacon devises are like lighthouses that send light signals to ships so they could know where they are and take actions based on their locations.

Figure 3.3 How iBeacons Work adopted from [estimote.com]
The iBeacon protocol makes use of the data field in a BLE packet in order to transmit its advertisement packet. Figure 3.4 describes the structure of this packet.

![Image of iBeacon advertisement packet](image)

Every iBeacon advertisement packet contains iBeacon prefix, iBeacon ID and transmission power (TX).

The iBeacon Prefix contains the following:

- 3 bytes advertisement flag (0x020106) defines the advertising packet as BLE general discoverable and BR/EDR incompatible. In other words, this is only broadcasting and non-connectable.
- 2 bytes advertisement header (0x1AFF) says the flowing data is 26 bytes and manufacturer specific data.
- 2 bytes company ID is the start of the manufacturer specific advertising payload. The Bluetooth specification requires that manufacturer specific payloads begin with the company ID number. Since iBeacon is Apple’s proprietary, Apple’s company ID is used (0x4C00).
- 1 byte iBeacon type (0x02) it is fixed and used by all iBeacons.
- iBeacon length (0x15) that defines the remaining length to be 21 bytes.

The iBeacon ID is 20 bytes length used in order to identify a specific beacon within a region. It is divided into three sections:
• 16 bytes Universally Unique Identifier (UUID) are defined by the application developers. Typically, it is unique to a vendor and application.

• 2 Bytes Major number further specifies a specific iBeacon and use case for example this could define a sub-region within a larger region defined by UUID. Major can take any unsigned integer value from 0 to 65536.

• 2 Bytes Minor number allows further subdivision of a region or use case. Minor can take any unsigned integer value from 0 to 65536.

These values are determined and configured by the person or the organization deploying the beacon device and don’t have to be registered with Apple.

The last byte in the iBeacon advisement contains the transmission power. This is a factory calibrated read only constant, which indicates the RSSI value at the distance of 1 meter away from the beacon in an obstacle-free environment. By knowing the expected RSSI and the actual RSSI measured by the receiver, it is possible to estimate the distance.

Table 3.1 shows examples of how these values may be used. A retail store wants to place iBeacon inside their stores in three different cities. A single application is used to recognize each store and its departments. Therefore, the UUID is shared by all locations so the application knows that the same retailer owns that store. Each specific store, Denver, Colorado Springs, and Boulder is assigned a unique major value allowing a device to identify which specific store it is in. Within each individual store departments are given separate minor values. However, these are the same across stores since each location has the same three departments.
Table 3.1 UUID, Major, and Minor example

<table>
<thead>
<tr>
<th>Store Location</th>
<th>Denver</th>
<th>Colorado Springs</th>
<th>Boulder</th>
</tr>
</thead>
<tbody>
<tr>
<td>UUID</td>
<td>B9407F30-F5F8-466E-AFF9-25556B57FE6D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Major</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Minor</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clothing</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Furniture</td>
<td>20</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>Jewelry</td>
<td>30</td>
<td>30</td>
<td>30</td>
</tr>
</tbody>
</table>

Using this information, a device that supports iBeacon could identify when it has entered or left one of the stores, which specific store it is at, and what department the user might be standing in.

By using Apple provided iBeacons APIs, location could be reported to an application through two functions: monitoring and ranging.

**Monitoring in Location Reporting**

Monitoring is a high-level view of whether both the beacon and the mobile device are loosely within the same space, often called a region. A device is either in the region when it is able to receive iBeacon, or it is out of the region when it unable to receive iBeacon. Consequently, an action on the device would be triggered by an application when the device enters or leaves a region. However, Region monitoring is limited to 20 regions and can function in the background of the listening device even if the device is locked.

In order for an application to define a beacon region, it must identify the UUID of the iBeacon advertisement to match. In addition to the UUID, an application can optionally define the major and minor values to further specify a beacon region to be monitored. In our retail store example, if the application only specifies A UUID for the beacon region then it will be notified when the user enters or leaves any of the stores. However, if the user
only wants to be notified when entering the store located in Denver then the application could use the UUID and major values in order to recognize the beacon region. Likewise, if the user is interested in being notified when he/she have entered a specific department in the store for example the jewelry department in Boulder, the application would use the UUID, major, and minor values to define the beacon region to match and monitor.

In brief, Monitoring function enables users to detect movement in and out of a region. However, doesn’t provide the distance from a specific beacon. For such requirement ranging function should be used.

**Ranging in Location Reporting**

Ranging function provides a list of beacons detected in a giving region along with their properties and the approximate distance between a beacon and a device. However, ranging operation works only in the foreground when the device is unlocked and doesn’t take in consideration any reflection or obstruction between the device and the beacon.

There are two parameters that could be used under the ranging function, proximity states and accuracy. The iBeacon API can estimate the following four proximity states:

- **Immediate**: indicates that device is physically very close, within 0.5 meter.
- **Near**: with a clear line of sight between the device and the beacon, this would indicate proximity between 0.5 to 2 meters. However, if there is attenuation of the signals due to the obstructions between the device and the beacon, this state may not be reported even though the device is in the range.
- **Far**: In this state, the confidence in the accuracy is too low to determine if the detected device is Near or Immediate. When Far is indicated, the accuracy property could be used to determine the potential proximity to the beacon, which could be a distance between 2 to 30 meters.
- **Unknown**: This state indicates there is no proximity can be determined. This could
be because the ranging function has just begun, insufficient measurement to
determine the state, or the device is more than 30 meters away from the beacon.

Figure 3.5 shows those four different proximity states. Notice that the proximity state
parameter doesn’t give distance. In addition, you might have multiple beacons return the
same proximity values, which won’t be precise enough if you need to get more specific.
Therefore, Apple’s API provides the developers with accuracy parameter to use.

![Figure 3.5 Four Ranging Proximity States adopted from [Koning 2014]](image)

Accuracy value provides the distance in meters and is calculated by using the strength
of the signal from the iBeacon. Due interference, some unknown values might be reported
as negative values.

Finally, Android doesn’t have native iBeacon support. Therefore, developers either
have to use third party library or write code that parses BLE packets to find iBeacon
advertisements. In addition, by design the iBeacon advertisement frame is not encrypted or
authenticated. Therefore, the iBeacon ID (UUID, major, and minor) can be spoofed and
assigned to a pirate beacon, which could be a security threat to this protocol.
3.3.2.2 Eddystone

Google announced its proximity location protocol in July 2015. It was called UriBeacon for a while then they changed the name to Eddystone.

Unlike iBeacon, Eddystone is an open protocol that transmits four different types that work with iOS and Android instead of one. These frames are:

**Eddystone-UID**

It is a unique identifier that can identify a particular beacon. Just like with iBeacon, an application installed on a phone can use the identifier to trigger desired action. Eddystone-UID is 20 bytes long and split into the following fields:

- 1 byte frame type: in all Eddystone frames this field indicates the frame type. UID frame type has value (0X00)
- 1 byte TX power: like iBeacon this field holds the calibrated power. However, this measured at 0 meters instead of 1 meter.
- 10 bytes namespace ID: similar in purpose to iBeacon UUID. Namespace is used to identify an entity.
- 6 bytes instance: similar in purpose to iBeacon Major and Minor. This field is used to identify a specific beacon.
- 2 bytes are reserved for future use.

**Eddystone-URL**

This frame is used to broadcast a URL that can be received by users proximity within the range of a beacon. With iBeacon and Eddystone-UID there is a need for an application to read the beacon’s identifier and translate it into desired action. However, with Eddystone_URL the data is encoded directly in the beacon advertisement. The means instead of having different application fetching contextual data, usually in from a website,
a web browser could be used. This frame forms the backbone the Physical Web [Google Physical Web].

Eddystone-URL is from 6 to 20 bytes long depends on the URL size and split into the following fields:

- 1 byte frame type with (0X10) value
- 1-byte TX power
- 1 byte URL scheme prefix: could holds the values (0X00) for http://www., (0X01) for https://www., (0X02) for http://, or (0X03) for https://
- 3 to 17 bytes encoded URL: contains a compressed web URL that can be launched using any physical web browser for example Google Chrome for mobile phones.

**Eddystone-TLM**

This frame is used to broadcast telemetry data in order to monitor the health and operation of a beacon. This telemetry data includes the battery voltage, beacon temperature, advertising packet counter since power up or rebooting, and beacon uptime.

Eddystone-TLM is 14-byte size and split into the following fields:

- 1-byte frame type with (0X20) value.
- 1-byte TLM version with (0X00) value.
- 2 bytes battery voltage, which could be used to estimate the battery level in millivolts.
- 2 bytes beacon temperature in Celsius sensed by the beacon. If not supported the value should be set to 0X8000, -128C.
- 4 bytes advertisement PDU count: the running count of advertisement frames of all types emitted by the beacon since power-up or reboot.
- 4 bytes time since boot: is a 0.1 second resolution counter that represents time since power-up or reboot.
Because this packet doesn’t contain any beacon ID, it must be paired with either UID or URL.

**Eddystone-EID**

In 2016 Google introduced Eddystone-EID, which is designed for security. Rather than using fixed namespace and instance ID, it broadcasts an encrypted ephemeral identifier that changes periodically and can be resolved by the service with which it was registered.

<table>
<thead>
<tr>
<th></th>
<th>Apple’s iBeacon</th>
<th>Google’s Eddystone</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Technology</strong></td>
<td>It is a proprietary software</td>
<td>It is open-source. The specification is published on GitHub under the open-source Apache v2.0 license</td>
</tr>
<tr>
<td><strong>Compatibility</strong></td>
<td>It is iOS and Android compatible, but native only for iOS.</td>
<td>It is Android and iOS compatible</td>
</tr>
<tr>
<td><strong>Broadcasted packets</strong></td>
<td>iBeacon broadcasts only one advertising packet</td>
<td>Eddystone broadcasts three different packets: Eddystone-UID, Eddystone-URL, and Eddystone-TLM</td>
</tr>
<tr>
<td><strong>Usage</strong></td>
<td>Mobile App is mandatory</td>
<td>Mobile App is optional when Physical Web is used</td>
</tr>
<tr>
<td><strong>Security</strong></td>
<td>None</td>
<td>Eddystone has a built-in feature called EID</td>
</tr>
<tr>
<td><strong>Management and Cloud Integration</strong></td>
<td>None</td>
<td>Google Proximity Beacon API Google Nearby API</td>
</tr>
</tbody>
</table>

In order to use this frame, beacons have to be registered with Google and Google Proximity API. It is 10 bytes frame contains:

- 1 byte frame type with (0X30) value.
• 1 byte TLM version with (0X00) value.
• 8 bytes ephemeral identifier

3.3.2.3 Manufacturer Specific Custom Profiles

Beside iBeacon and Eddystone, BLE protocol is flexible enough to allow manufacturers to configure customized BLE profiles for specific usage. Manufacturers can add extra information they need to the beacons; for example, battery voltage level, sensor measurement or whatever data they need to broadcast. However, The Manufacturer Specific Data Flag in the broadcasted PDU has to be set to “FF”. In addition, the PDU size has to be within the standard advertising PDU size. Finally, the application side needs to be redesigned for retrieving the correct data from the customized beacon packet.

3.3.3 Bluetooth 5

Bluetooth 5 was introduced on December 2016 as the latest revision of Bluetooth Core specification with the 2x Speed, 4x range and 8x advertisement data increase compared to Bluetooth 4.0

In order to increase the speed and the range, Bluetooth 5 supports three Physical Layer (PHY) modes compared to one in Bluetooth 4:

• **LE 1M PHY**: 1 Mb/s bit rate, modulated at 1 Megasymbol/sec symbol rate. Each bit maps to a single radio symbol. The same PHY as used in Bluetooth 4.0.
• **LE Coded PHY**: 500 or 125 kbit/s bit rate, 1 Megasymbol/sec symbol rate. Each bit is coded into 2 or 8 bits using a forward error correction algorithm to provide higher tolerance for bit errors, resulting in improved range.
• **LE 2M PHY**: 2 Mb/s bit rate, 2 Megasymbol/sec rate. Doubles the symbol rate to increase the speed, at a small cost to range.
There are certainly trade-offs for choosing one of these PHYs over the other. For example, the new LE 2M PHY can increase the speed and also reduce the power consumption since the same amount of data is transmitted in less time. On the other hand, the benefit of using the LE Coded PHY is increase range with the trade-off of both higher power consumption and reduced speed.

In addition, according to the Core Specification section Volume 1, Part A 1.2 both LE Coded PHY and LE 2M PHY are optional. In other words, vendors can claim Bluetooth 5 support, but they won’t necessarily support the faster speed and longer-range improvements.

### 3.3.3.1 Bluetooth 5 Extended Advertisement

In Bluetooth 4.0, the advertising payload was a maximum of 31 bytes. In Bluetooth 5, the payload is increased to 255 bytes by adding additional advertising channels and new advertising PDUs.

Recall in Bluetooth 4.0 all advertising was done on 3 of the 40 band channels (37,38, and 39). However, with Bluetooth 5 there are two sets of advertising channels: primary and secondary.

The primary advertising channels are the original 3 channels defined in Bluetooth 4. While the secondary advertising channels use the 37 fixed previously reserved for data.

Advertisements always start with advertainment packets sent on the primary channels. When larger packets need to be sent, this extra information can then be offloaded to the secondary channels to allow for more data to be broadcast. This is shown in Figure 3.6.
If 255 bytes is not enough for some cases that require even larger amount of data to be broadcast, it’s possible to chain packets together and for each packet to contain a different subset of the whole data set. Each chained packet can be transmitted on a different channel with a referencing header to the next in the chain. This is shown in Figure 3.7.

Another feature of Bluetooth 5 Extended Advertisements is **periodic advertising**. With Bluetooth version 4 to help avoid persistent packet collisions, advertising usually includes a degree of randomness inserted in the advertising event scheduling process. However, Bluetooth 5 introduces the ability to perform periodic advertising, which allows...
scanners to synchronize their scanning for packets with the advertising devices. Bluetooth 5 now defines a synchronizable and non-synchronizable modes. When operating in synchronizable mode, a Periodic Advertising Synchronization Establishment procedure is defined and a new header field in the Extended Advertisement header called SyncInfo, which contains timing and timing offset information is leveraged [BLE Core Specification]

### 3.3.3.2 New Extended Advertising Protocol Data Unit (PDU)

In order to provide the new Extended Advertisement features, Bluetooth 5 has eight different types of advertising packets that can be grouped into two categories:

1. Legacy Advertisements: Same advertisements from Bluetooth version 4. They include the following PDU types
   - ADV_IND
   - ADV_DIRECT_IND
   - ADV_NONCONN_IND
   - ADV_SCAN_IND

2. Extended Advertisements: These can be used with the new Extended Advertisement features. However, they can only be discovered by devices that support these features. They include the following PDU types:
   - ADV_EXT_IND
   - AUX_ADV_IND
   - AUX_SYNC_IND
   - AUX_CHAIN_IND

Table 3.3 explains the different advertising PDUs:
Table 3.3 Different advertising PDUs

<table>
<thead>
<tr>
<th>Advertising PDU</th>
<th>Description</th>
<th>Channel</th>
<th>Allow scan req</th>
<th>Allow connect</th>
<th>Permitted PHYs</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADV_IND</td>
<td>Used to send connectable undirected advertisement</td>
<td>Primary Adv</td>
<td>Yes</td>
<td>Yes</td>
<td>●</td>
</tr>
<tr>
<td>ADV_DIRECT_IND</td>
<td>Used to send connectable directed advertisement</td>
<td>Primary Adv</td>
<td>No</td>
<td>Yes</td>
<td>●</td>
</tr>
<tr>
<td>ADV_NONCONN_IND</td>
<td>Used to send non-connectable undirected advertisement</td>
<td>Primary Adv</td>
<td>No</td>
<td>No</td>
<td>●</td>
</tr>
<tr>
<td>ADV_SCAN_IND</td>
<td>Used to send scannable undirected advertisement</td>
<td>Primary Adv</td>
<td>Yes</td>
<td>No</td>
<td>●</td>
</tr>
<tr>
<td>ADV_EXT_IND</td>
<td>Used to indicate that an advertisement will be sent on a secondary advertisement channel.</td>
<td>Primary Adv</td>
<td>No</td>
<td>No</td>
<td>●</td>
</tr>
<tr>
<td>AUX_ADV_IND</td>
<td>Used to send connectable directed advertisement on a secondary advertisement channel</td>
<td>Secondary Adv</td>
<td>Yes</td>
<td>Yes</td>
<td>●</td>
</tr>
<tr>
<td>AUX_SYNC_IND</td>
<td>Used for periodic advertisements on secondary advertisement channels.</td>
<td>Secondary Adv</td>
<td>No</td>
<td>No</td>
<td>●</td>
</tr>
<tr>
<td>AUX_CHAIN_IND</td>
<td>Used to chain advertisement packets, allowing the advertisement data to extend beyond 255 bytes.</td>
<td>Secondary Adv</td>
<td>N/A</td>
<td>N/A</td>
<td>●</td>
</tr>
</tbody>
</table>

Notice that all of the legacy advertainment PDUs can only be sent on the original LE 1M PHY and all Secondary Advertising PDUs can be sent on any of the three PHYs.

Let’s give examples the use of these new PDUs. If you want to advertise on LE Coded PHY (for long range), the device would use ADV_EXT_IND to advertise on the primary advertising channels. The ADV_EXT_IND PDU contain a pointer, AuxPtr field, to a AUX_ADV_IND PDU that would be transmitted on a secondary advertisement channel.

Another example; if the broadcasting device wants to send more than 255 bytes payload. It can broadcast with ADV_EXT_IND, which contain a pointer to AUX_ADV_IND. The AUX_ADV_IND could be sent on the LE 2M PHY for speed. The AUX_ADV_IND contain the AuxPtr pointer to an AUX_CHAIN_IND which contains the remanding of the advertising data. Notice in this case, the device is not allowed to advertise in connectable or scannable mode.

Bluetooth 5 with the higher symbol rate of LE 2M and advertising extinctions features will have substantial impact on the next-generation beacon technology. However, there will be a wait involved until relevant devices such as smartphones, computes and developer kits for researchers that fully support the new features will be in the market it’s a wait that will be worthwhile.
Chapter 4

Attacks on BLE Beacon based Indoor Position Systems

4.1 Introduction

Bluetooth Low Energy (BLE) beacon provides a means for associating a specific location with a particular Bluetooth signal. It is a technology that allows any device with BLE capability to listen to the beacon signals from the beacon transmitters and react accordingly. BLE beacon could help the mobile devices to compute their position and facilitate the delivery of the location-based content. For example, consider a BLE beacon device installed inside a coffee shop. When a customer enters the coffee shop, the signal transmitted from the beacon will help the coffee shop application installed on the customer phone to determine it has entered the coffee shop. Then, the application could show a welcome message or connect to a local or remote server to download the promotion items or the latest product list.

BLE technology can do more than just help shopping, it can be used for museum guides, attendance recording, provide information about physical space to help those with visual impairments, and more. In other worlds, this technology is useful anytime an application needs to deal with location or interact with the physical world.

BLE beacon device is a transmit only device. It broadcasts at regular intervals an advertising packet containing ID used to identify a particular beacon. The beacon protocol also includes a field called tx-power that allows the receiver to calculate its relative distance to the beacon device. However, because of the transmitted packet size of the current 4.2 Bluetooth version is limited to 32 bytes, a beacon device cannot cryptographically sign any
broadcasted packets. Hence, the receiver cannot authenticate the identity of the transmitter. This limitation may lead to different attacks against the BLE beacon infrastructure. For instance, someone with a BLE sniffer application can obtain the beacon ID and then spoof it by creating another beacon with the same ID.

This section presents five attacks that can be launched against BLE beacons. We conducted experiments in the lab and in a public mall to collect the related BLE beacon data and verify these attacks.

Through the study we have identified a new attack, called hide and seek attack, whereby changing the tx-power field value, the attacker can mislead the normal users to incorrect geolocations and also avoid being located.

### 4.2 Contribution

- Identify the hide and seek attack on the BLE beacon geolocation system.
- The data set collected from the lab and the public mall and their analysis.

### 4.3 Attack Threats Against BLE Beacons

BLE beacons has two methods to communicate with the outer world. First method is a two-way communication used by the administrator for configuration, which is encrypted and protected by username and password. The second is the one-way broadcast to transmit the advertising packet to receivers, which is unencrypted. Consequently, most attacks on beacon devices happened as a result of attacking the unencrypted communication. The attacks on the BLE beacon can be identified as follows:
4.3.1 Piggybacking/free-ride attack

it is a kind of abuse where an unauthorized party uses the beacon infrastructure without prior consent of the infrastructure owner. Imagine that Alice running her own bakery shop. She uses BLE beacon technology to push personalized offers to her customers. Next door store owner Eve wants to piggyback on those beacon packets. Eve can do that by sniffing Alice’s beacons IDs and use them in Eve own application. Then, every time someone with Eve’s application installed visits Alice’s store, they are exposed to push notifications about the deals at Eve’s store. Sharing infrastructure for free with strangers is inconvenient. However, this attack doesn’t damage the beacon infrastructure, the victim’s application or has a negative effect on the business owner customers.

4.3.2 Spoofing/cloning attack

It is an act where the advertisement packet of a BLE beacon is cloned to a different beacon device in order to impersonate or spoof the original beacons. This attack could harm the beacon infrastructure owner since it could enable beacon infrastructure service outside its service area. Imagine a hospital where doctors have to be within the hospital area in order to access the patients’ medical records. To accomplish this, the hospital management installed a location verification system based on BLE beacon technology. Doctors have to use their location application to prove their presence by reporting the surrounding beacon devices. To attack that system, an attacker needs to install the location application, scan the beacons inside the hospital, getting their IDs, and creating other beacons with the same IDs. Now the attacker can place the cloned beacons anywhere he wants even outside the hospital. However, the location application will verify his location is inside the hospital.
4.3.3 Packet injection attack

This attack is very similar to beacon spoofing. However, instead of placing the cloned beacon outside the service area, the attacker places the cloned beacons among the original network in order to disturb its normal operation. For example; a museum uses BLE beacons for their guide application. Someone could clone the beacons at the Asian Art section and place them inside the Contemporary Art section. Therefore, a visitor inside the Contemporary Art section will receive information not meant to be received at that section.

4.3.4 Hide and seek attack

The goal of this attack is to manipulate a BLE receiver into sensing a beacon broadcaster as closer or farther away from its actual location.

Recall that BLE beacons include in their advertising a transmitter power that indicate how strong the signal should be at a known distance. This distance usually is at one meter. On the receiver device each time a beacon advertisement packet is received, the Bluetooth chipset provides a measurement of the received signal level as Received Signal Strength Indicator (RSSI).

Because every single beacon transmission also includes the calibration value mentioned above, it is possible to compare the actual signal level with the expected signal level at one meter and then estimate the distance. For example, let’s say the beacon advertisement packet was received with a signal level of -65 dBm and the transmitter power calibration value sent inside the transmission was -59 dBm. Because -65 dBm represents a weaker signal level than -59 dBm (greater negative numbers represent weaker signals) this
means the beacon is probably more than one meter away. On the other hand, if the advertisement packet was received a signal stronger than -59 dBm, this means the beacon device is less than one meter away from the receiver.

For attackers to launch a hide and seek attack, they may introduce their own equipment that broadcasts faked calibration values which could result in an inaccurate proximity estimation. In our test, we used a program written in python and running on Raspberry Pi model 3B in order to broadcast iBeacon advertisement packet with customized payload. We place our device one meter away from an iPhone running iBeacon scanning program. Within one meter, the power calibration value is -46dBm. However, we broadcasted two misleading calibration values, -90 dBm and -20dBm. In the -90 dBm case the iPhone device measured distance less than a meter. Figure 4.1(a) shows the output of the scanning program. However, when -20dBm is broadcasted, iPhone device measured distance more than 12 meters. Figure 4.1(b) shows the output of the scanning program.

![a) b) Figure 4.1 Output of the scanning program](image-url)
4.3.5 Physical attack

Beacon devices can be stolen, moved from one location to another, or a pair of beacon devices could be switched their position by an attacker. All of those attack actions will mess up the beacon’s advertisement distribution and disturb the service that a beacon infrastructure provides.

4.4 Case Study

In this case study we conducted experiments to measure the security of one of the real-life iBeacon implementations at one of the shopping centers in Colorado. The mall offers its visitors a mobile application that can be downloaded from Apple store for free in order to provide indoor shortest route navigation between the stores and sales promotions services. However, these features are only available inside the malls, as shown in Figure 4.2.

The goals of our experiments are to enable these features without being inside the shopping center. In addition, we want to test if we can launch a geolocation spoofing attack in order to mislead the shopping application and appear to be located at a location that we want within the mall without being actually at that location.
4.4.1 Experiment 1: Attacking the location restrictions

In this experiment we launch spoofing/ cloning attack against the mall iBecaon infrastructure in order to enable the mall application features outside the mall.

The attack can be divided into the following steps:

- Scanning the iBeacon infrastructure.
- Collecting beacon devices IDs.
- Cloning one of the beacons.

We used BeaconTools 1.3.0, python library that uses Linux Bluetooth protocol stack, and python 3.6 language to write a BLE beacon scanning program on a Raspberry Pi 3B model device that comes with built-in BLE. We used this scanning program to collect some of the installed beacons IDs. Figure 4.3 shows the results of the scanning program.
Later on, while we were outside the mall area, in another city, we used another program we written in python 3.6 to clone one of the mall’s beacon and broadcast the same iBeacon ID from our Raspberry Pi device. The command is shown in Figure 4.4.

We launch the shopping center application and as a result, the application assumes that we are inside the mall and we can use all of its functions without any limitations. Figure 4.5 shows the capture images of the mobile app.
4.4.2 Experiment 2: Geolocation spoofing attack

In this experiment we choose a location within the mall where we want to pretend at. The goal of this experiment is to mislead the application to believe that we are at that location even though we are outside the mall.

This attack can be divided into the following steps:

- Scanning the iBeacon infrastructure around the target area.
- Collecting beacon devices IDs around the target area.
- Cloning three of the selected beacons.

We used our scanning program to build a database of the surrounding beacon devices. In the database we include the beacon IDs, the signal strength, and the tx-power of those devices. This information can be used to estimate the distance between our location and the surrounding beacons. However, we found out that this attack can be launched even without knowing the location of any of the beacon devices. Assuming that
we receive more broadcast signals from near beacon devices and less from farther ones, we just need to find in the collected database the devices that broadcast more than others.

In our testing lap after analyzing the database, we selected the most frequent three beacon IDs. We used our cloning program on 3 Raspberry Pi 3B model devices to broadcast the three selected IDs. We launched the shopping center application as a result the application map shows that we are at the target location as seen in Figure 4.6 even though we were at different city. This shows that launching a geolocation spoofing attack against BLE infrastructure is doable.

![Figure 4.6](image)

a) The application showing our location inside the mall. b) the application showing our location after spoofing our geolocation from different city.

### 4.5 Conclusion

In this chapter we present five attacks that can be launched against BLE beacons. Identify the hide and seek attack on the BLE beacon geolocation system. In addition, we present a case study where we conducted experiments to measure the security of one of the
real-life iBeacon implementations. The prototypic implementation uses off the shelf embedded devices which is low cost and small size. The experiment results demonstrate the feasibility of the attacks and the need for security system to monitor and prevent such attacks.
Chapter 5

Selecting Filters for BLE Beacon Attack Detection

5.1 Introduction

In this section, RSSI measurements are analyzed, different RSSI filters are applied and compared in order to provide better RSSI prediction.

5.2 RSSI Analysis

In order to analyze the RSSI values variation in the tested environment, 500 RSSI values are measured with a constant distance of one meter away from the BLE beacon. Figure 5.1(a) shows the measured RSSI variance within an interval of 29 dBm and an average of -59 dBm. The histogram in Figure 5.1(b) displays RSSI values of all measurements and shows that the distribution of the RSSI values spreads around -61 dBm. Taking in considering that the broadcasted tx-power, which indicates what the expected RSSI should be at one meter as calibrated by the BLE beacon manufacture, is -64 dBm
This signal fluctuation could happen due to the following characteristics: **signal strength attenuation**, **signal interference** and **multipath propagation**. As we previously mentioned, the attenuation occurs when the BLE signal becomes weaker as they pass through objects or travel longer distance between a transmitter and a receiver. **Signal interference** happens when the transmission signal cannot be extracted at the receiver due to the interference between the transmitted signal with another identical frequency bandwidth. Finally, **multipath propagation** occurs when BLE signals reach the receiver.
through multiple paths of different distances as they bounce against objects in the environment such as wall and furniture [Huh2017].

These results show that the raw RSSI values are not reliable enough and some RSSI filters are required to improve the stability of the RSSI values before they are used in our system.

5.3 RSSI Filters

Due to the significant variance in the RSSI measurements, it is necessary to filter and smooth the RSSI values. In this section, the moving average, moving median, and Kalman filter are compared in order to find the best fitting filter for the RSSI measurements.

5.3.1 Moving Average and Moving Median Filters

The average filter works by defining a fixed size sliding window that moves through the data and replace each value with the average value of all values within the window [Smit1999]. On the other hand, the median filter works by replacing each value with the median value of all values within the predefined sliding window [Aria2009].

5.3.2 Kalman Filter

In this section, we cover the high-level operation of Kalman Filter. After representing the broad overview, we will narrow the focus to the implementation of Kalman Filter as RSSI smoothing technique.

The Kalman Filter is introduced by Rudolph Kalman, who in 1960 published his paper titled “A New Approach to Linear Filtering Prediction Problems” [Kalman 1960]. In his paper, he describes a recursive solution that could be used to estimate the state of dynamic
systems from a set of process measurement taking into consideration the inaccuracy and noisy nature of these measurements.

Note that state is a list of the data that we want to keep track of or estimate its true values. For example, it could be the position and velocity of an object, the amount of fluid in a tank, the room temperature or any number of parameters that we need to guess. In our case, it is the RSSI values. In addition, depends on the number of the states in the system that we want to estimate, Kalman Filter could be defined as n-dimension Kalman Filter, where n is the number of states. i.e. in the position and velocity example we have 2D-Kalman Filter while measuring the room temperature is 1D Kalman Filter.

Kalman Filter has been widely applied in the fields of navigation, tracking objects and positioning. In addition, Kalman Filter has proved useful in the field of computer vision applications such as digital image processing, pattern recognition, image segmentation and image edge detection. [Qiang 2015]

The algorithm works in two-step recursive process, **Time Update (prediction)** and **Measurement Update (correction)**. In the prediction step, the Kalman Filter projects the current state variables, along with their error covariances. In the correction step, the next noisy observed measurements are incorporated into the priori estimation to calculate new improved posteriori estimations along with their new updated error covariances, which will be used as a feedback for the next prediction step.

The following Figure 5.2 provides a low-level schematic description of the algorithm.
Kalman Filter algorithm consists of two sets of equations, **Time Update (prediction)** and **Measurement Update (correction)** equations. Both sets are applied at each $k^{th}$ state or time step. The general forms of Kalman Filter equations are as below:

1) **Time Update (Prediction) Equations**

$$\hat{x}_k^- = A \hat{x}_{k-1} + B u_k$$

$$P_k^- = AP_{k-1}A^T + Q$$

2) **Measurement Update (Correction) Equations**

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$$

$$\hat{x}_k = \hat{x}_k^- + K_k (z_k - H \hat{x}_k^-)$$

Figure 5.2 The ongoing discrete Kalman filter cycle. The time update projects the current state estimate ahead in time. The measurement update adjusts the projected estimate by an actual measurement at that time.
\[ P_k = (I - K_k H) P_k^- \]

Where

\( \hat{x} \) : Estimated State

\( A \) : State transition matrix (i.e. update x and P based on the time that has passed)

\( u \) : Control variable matrix

\( B \) : Control matrix (i.e. relates the control variable matrix u to the state \( \hat{x} \))

\( P \) : Process covariance Matrix or Error covariance (i.e. represents the error in the estimate)

\( A^T \) : The matrix transpose of A

\( Q \) : Process noise covariance matrix (i.e. error due to process)

\( K \) : Kalman gain (i.e. weight factor based on comparing the error in the estimate to the error in the measurement, \( 0 < k < 1 \))

\( H \) : Measurement matrix (i.e. it helps transform the matrix format of P into the format desired for the K matrix)

\( H^T \) : The matrix transpose of H

\( R \) : Measurement noise covariance matrix (i.e. error due to measurement)

\( z \) : Measurement vector (i.e. the noisy measurement)

\( I \) : The identity matrix.

In the above equations, superscript “-“ denotes the a priori value. for example, \( \hat{x}_k^- \) is defined as a priori estimated state at time k, and \( P_k^- \) is the priori error covariance. These priori values are used in the correction equations.
The first step in the correction equations is to calculate Kalman Gain $K$, which will be used to update the posteriori estimation $\hat{x}_k$ via $z_k$. In addition, it will be used to update the posteriori $P_k$. These updated values will be used as priori $\hat{x}_k^-$ and $P_k^-$ in the next iteration.

The Kalman Gain equation can be simplified as below:

$$K_k = \frac{error\ in\ estimation}{error\ in\ estimation + error\ in\ measurement}$$

What the Kalman Gain does is putting a relative importance between the error in estimation and the error in measurement. In other words, which value should be trusted more in the calculation, the estimated or the measured value.

If Kalman Gain is close to 1, that means we have accurate measurement with minimum error. Therefore, the posteriori estimation $\hat{x}_k$ will be strongly updated based on the measurement $z_k$. However, if the Kalman Gain is small, then the error in the measurement is larger than the error in the estimation. Consequently, the difference between the measurement and $z_k$ and the priori estimated state $\hat{x}_k^-$ will have a smaller effect on $\hat{x}_k$ calculation.

**5.3.2.1 Kalman Filter for RSSI smoothing**

In our implementation, we use RSSI values between the BLE beacons and the monitoring device as the state that we want to estimate its value, $\hat{x}$. Since we are tracking one state, we are adopting 1D Kalman Filter implementation. Therefore, every entity in our Kalman Filter equations is a numerical value, not a matrix. Furthermore, $z_k$ will be the RSSI values that we measured at the monitoring device. In addition, in our implantation we make the following assumptions based on our observation:
• We don’t have control input u. Therefore, $B \cdot u_k$ is set to zero.
• The RSSI value measured at the monitoring device should be constant values since the BLE beacon will be fixed in normal saturation. Therefore, $A$ is just 1 because we already know the next value will be the same as the previous one.
• Since we use 1D Kalman Filter, $I$ and $H$ is one.
• We assume that most of the noise is caused by the measurement, not from the processing. Therefore, we will use small value for $Q$.

Based on previous assumptions, Kalman Filter equations for our implementation are the following:

1) Time Update (Prediction) Equations

$$\hat{x}_k^- = \hat{x}_{k-1}$$

$$P_k^- = P_{k-1} + Q$$

2) Measurement Update (Correction) Equations

$$K_k = P_k^- (P_k^- + R)^{-1}$$

$$\hat{x}_k = \hat{x}_k^- + K_k (z_k - \hat{x}_k^-)$$

$$P_k = (1 - K_k)P_k^-$$
In our implementation we used $\hat{x}^c_k = 0$ and $P^c_k = 1$ as initial values for the first iteration. For Q we used $Q = 0.0001$ as small number. To determine R value, we calculated the absolute error between the broadcasted BLE beacon TX power, which indicates what the expected signal level should be when you are one meter away, and the measured RSSI one meter away from the beacon device. Form the experiment, we found $R = 7$.

5.4 Filters Evaluations

This section elaborates on the experimental methodology we adopted for evaluating the performance of these. First give a description of the method, followed by our experiments and the evaluation results.

5.4.1 Experiment Design and Methodology

We have conducted our experiments inside a 10m by 8 m personal apartment comprising furniture and electrical equipment. We installed five BLE beacons and a monitoring device within the environment three meters above the floor to minimize human interference as shown in Figure 5.3.
For the data gathering phase and monitoring device we used Raspberry Pi 3b model device with a BLE beacon scanning program written in python 3.6 with BeaconTools 1.3.0 library. The beacons transmission power is set to 0dBm to guarantee that the five beacons signals could cover the exterminate area. In addition, the advertisement interval set to 100ms as recommended by Apple. We used the widely used python library Scikit-learn in our implementation to train and test our machine learning models.

For every BLE packets received by the monitoring device, we logged the respective unique ID, RSSI values and the received time. We conducted four runs of this data gathering phase in order to collect four different datasets for 5 minutes. First run to collect
raw data without applying any filtering algorithm and the other three runs for the filtered data. In the first run, we collected 10490 packets within 5 minutes scanning window. For the other three runs, we divided the 5 minutes window into smaller 5 seconds scanning windows. During these 5 seconds the monitoring device can receive 40 packets in average from each installed BLE beacon. By the end of each scanning window, we apply the filtering algorithms and record the filtered RSSI values. For moving average and median we used filtering window of 10 packets and for Kalman filter we recorded the filtered value after applying the algorithm on the collected 40 packets. The dataset sizes for the filtered RSSI values were 5799 packets for moving average and median filters and 1230 for Kalman filter. Figure 5.4 shows the Impact of the filtering process on the collected RSSI values from beacon1, beacons2, beacon3, beacon4 and beacon5.

![Figure 5.4 Impact of the Filtering Processes](image)

5.4.2 Selecting the Best Fit Filter Algorithm

In order to choose the best fit filter algorithm, we need to measure their performance when a machine learning algorithm is applied to predict the Beacon ID given the RSSI values under normal system operation.
We used the four datasets to train and test three different machine learning classifiers, 
**Support Vector Machine (SVM), Random Forest (RF), and Decision Tree (DT).** When 
training the different classifiers, we used 10-fold cross-validation method to make sure that 
each and every data point comes to test at least once. In 10-fold cross-validation methods, 
first, we shuffle the dataset randomly. Secondly, we split the dataset into 10 groups. Finally, 
for each unique group, we take the group as a test dataset, take the remaining groups as a 
training dataset, fit a model on the training set and evaluate it on the test set and retain the 
evaluation score.

In order to evaluate a machine learning model performance there are different metrics 
can be used. The most commonly used are accuracy, precision, recall, and F1score 
[Sokolova M, 2009]

Accuracy, which defined as the number of correct predictions that include the true 
positive (tp) and true negative (tn) predictions over the total number of predictions, could 
provide overall effectiveness of a classifier. However, it doesn’t take in consideration false 
negative(fn) and false positive (fp) cases. Accuracy can be formulated as the following

$$ \text{Accuracy} = \frac{tp + tn}{\text{total number of predictions}} $$

Precision is defined as the number of correctly classified positive cases divided by the 
number of the cases labeled by the system as positive. Precision can be formulated as the 
following:
Precision = \frac{tp}{tp + fp}

Recall is defined as the number of correctly classified positive cases divided by the number of positive cases in the data. Recall can be formulated as the following

Recall = \frac{tp}{tp + fn}

F1score is an overall measure of the machine learning model that combines precision and recall. F1score can be formulated as the following

F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}

F1 can have a value between 0 and 1. A value of F1 close to 1 indicates the best classification performance and meant that we have low false positive and low false negative predictions.

In our evaluation we used F1Score as our metric since it combines both precision and recall. In addition, we used confusion matrices to show how each classification algorithm performs. In the confusion matrix, each row represents the instances in an actual class. In our case the Beacon ID, while each column represents the instances in a predicted class. The elements along the diagonal represent the percentage of instances when the predicted Beacon ID is the same as the actual Beacon ID. While, the off-diagonal elements represent the mislabeled instances.
5.4.3 Filter Evaluation Results

Table 5.1 presents the average F1 scores for the trained classifiers with our four datasets and using 10-fold cross-validation method. It shows that Moving Median filter produced the best performance followed by Kalman Filter and then Raw Data and Moving Average.

<table>
<thead>
<tr>
<th>Filter Methods</th>
<th>SVM</th>
<th>RF</th>
<th>DT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw Data</td>
<td>0.77</td>
<td>0.77</td>
<td>0.77</td>
</tr>
<tr>
<td>Moving Median</td>
<td>0.91</td>
<td>0.91</td>
<td>0.91</td>
</tr>
<tr>
<td>Moving Average</td>
<td>0.75</td>
<td>0.76</td>
<td>0.76</td>
</tr>
<tr>
<td>Kalman</td>
<td>0.84</td>
<td>0.76</td>
<td>0.84</td>
</tr>
</tbody>
</table>

The confusion matrices in Figures 5.5 to 5.16 show the experimental results belonging to our tested machine learning classifiers and filtering methods.
Figure 5.5 Raw Date and SVM

Figure 5.6 Raw Data and Random Forest

Figure 5.7 Raw Date and Decision Tree

Figure 5.8 Moving Median and SVM
Figure 5.9 Moving Median and Random Forest

Figure 5.10 Moving Median and Decision Tree

Figure 5.11 Moving Average and SVM

Figure 5.12 Moving Average and Random Forest
By inspecting the confusion matrices, we can observe that the moving median is producing the best prediction with less mislabeling compared to the other filtering methods. Additionally, in Figures 5.8 to 5.16, the three machine learning algorithms provided similar prediction rate where beacon 1 is mislabeled as beacon 2 and beacon 5 is mislabeled as beacon 4.

Based on our results and observation, moving median filter will be used in our detection system.
5.5 Normal Operation Profiling Based on Moving Median Filter Evaluation

In this section we evaluate our machine learning model with moving median filter in predicting real-time new unseen legitimate BLE frames.

5.5.1 Experiment Design and Methodology

We used the same environment and setup in figure 5.3 as our testbed for our experiment. Our experiment involves two stages: the offline stage and the online stage. In the offline stage, the legitimate device profile is built. During profiling, the monitoring device scans the environment for a scanning window size of 5 seconds. During this window, RSSI values from each beacon will be collected and labeled with their beacon IDs. Those values will be filtered using moving median filter and used to build our profile dataset. We use the dataset to train the classifiers and store the model for predicting new packets. In the online stage, the monitor device will scan the environment for BLE frames for a scanning window size. During this window size, the RSSI values from each beacon will be collected and filtered using moving median filter to create datasets for each beacon discovered. These datasets will be fed to the machine learning model. The model then predicts if the packet comes from the same beacon ID or not.

In our experiment we used three different machine learning classifiers, **Support Vector Machine (SVM)**, **Random Forest (RF)**, and **Decision Tree (DT)** running on our monitor device. During the offline stage, we built a dataset contains 5100 filtered RSSI
values with their beacon IDs. For the online stage, we tested two scanning window size, 5 and 10 seconds for 200 tests.

In our evaluation we used the following metrics to compare among the three machine learning classifiers

- *F1score.*
- *Training time* required to train the machine learning model using the datasets collected during the offline stage.
- *Prediction time* required to predict the new unseen frames during the online stage.
- *False positive rate* is the cases which our model incorrectly predicts wrong labels.

### 5.5.2 Evaluation Results

Table 5.2 summarizes the results of our evaluation. It shows the evaluation metrics for our machine learning classifiers with 5 and 10 seconds scanning window size.

<table>
<thead>
<tr>
<th>Evaluation Metric</th>
<th>5 Seconds Scanning Window Size</th>
<th>10 Seconds Scanning Window Size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SVM</td>
<td>RF</td>
</tr>
<tr>
<td>F1Score</td>
<td>0.94</td>
<td>0.95</td>
</tr>
<tr>
<td>Training time in seconds</td>
<td>0.75</td>
<td>1.0014</td>
</tr>
<tr>
<td>Prediction time in seconds</td>
<td>0.0066</td>
<td>0.0571</td>
</tr>
<tr>
<td>False Positive Rate</td>
<td>4%</td>
<td>1%</td>
</tr>
</tbody>
</table>
We can observe from Table 5.2 while Random Forest classifier performed slightly better than the other classifiers, Decision Tree classifier has the fastest training and predicting time compared to Support Vector Machine and Random Forest classifiers. Additionally, all three classifiers have lower false positive rates using 10 seconds window size. This was to be expected as when the size of the scanning window increases, the more RSSI values are collected and filtered. However, there is a tradeoff between the scanning window size and the system responsiveness, as a larger window size would result in higher amount of time needed to collect the information required for each beacon and longer production time.

5.6 Conclusion

In this chapter we used our stationary monitor device to test and implement different filtering algorithms in order to minimize BLE signal strength attenuation. In addition, we evaluate different machine learning classifiers for real time prediction for legitimate BLE frames. The results show that our proposed system has f1 score of .95 with 1% false positive rate.
Chapter 6

BLE Beacon Detection

6.1 Introduction

In this chapter we propose a novel RSS-based detection system. We provide an overview of the architecture and a description of the system’s building blocks. In addition, we describe the attacks that could be detected by our system, the detection methods for each attack, experiments design to evaluate our methods and the results. Finally, we use the graph theory to build a framework that can be used to help positioning a monitoring device in the BLE infrastructure.

6.1.1 Contribution

1. A novel RSS-based detection module of BLE beacon geolocation system for detecting the malicious attackers.

2. The positioning monitoring framework based on the graph theory.

6.2 Architecture overview

The proposed solution, shown in Figure 6.1, is based on a stationary device placed inside the BLE beacon infrastructure to detect if a beacon is removed from its location or switched with another beacon. In addition, to identify if a hide and seek or packet injection attack are launched. The main components represented in the figure are:
1. The legitimate beacon set are the installed beacons in the infrastructure and need to be monitored.

2. The monitor device set that capture the BLE beacon frame and collect the corresponding RSSI values.

3. Detecting system that responsible of detecting misbehaved beacons and has the following modules:

   A. Detecting module that can be divided into two main components:

      **Supervised machine learning** based detection engine and **threshold-based** detection engine. The supervised machine learning based engine is responsible for detecting if a beacon is swapped with another beacon within the BLE beacon infrastructure. On the other hand, the threshold-based engine is used to detect packet injection, hide and seek attacks and if a beacon is moved from its original location.

   B. Localization module, the objective of which is to proximity find the distance between the monitor device and the cloned and moved beacons.
6.2.1 Legitimate BLE Beacon

These are the broadcasting devices that build the BLE infrastructure and need to be monitored. In order to identify the beacons individually, each device should be configured with its unique Identifier, which will be broadcasted as part of the transmitted packet along with the tx-power field.

6.2.2 Monitor Device

This device supports BLE and has to be capable of capturing BLE packets. It has two modes of operations, data gathering mode and monitoring mode. When the device is in data gathering mode, the device captures all BLE traffic within range and filters out advertising packets being transmitted from the infrastructure beacons. To do that, the device will check if the captured BLE packet confirms to the beacon protocol structure by looking for the protocol header. If this is found, the device will check for the advertising packets that has our beacons’ IDs. Then it creates a table contains a timestamp of the time the packet was captured, the RSSI value, tx-power, and the beacon ID. The information in this table will be used as input for our detection system. The data gathering mode has to run for set time frames and during normal non-attack situation.

In the monitoring mode, the device will capture all BLE traffic in range and filter out the beacons advertising data that the system will monitor. Similar to data gathering mode. Except in this mode the capture process of BLE advertisement is continuously running rather than for a set time frame. In addition, an attack on the BLE infrastructure could exist in this mode.
6.2.3 Detecting System

In our approach, RSS values are the signal strength of the beacon frames captured by the monitor device. Those values are closely related to the beacon physical location. Since the locations of these beacons from our monitor device are distinctive in the physical space, the RSS values of the beacon frames are also usually different. Therefore, we will use RSSI values as indication of the location of the BLE beacon in the infrastructure.

The **threshold-based** detection engine is useful against attacks that are clearly identifiable through thresholds and which are considered stable in the long term. This engine is responsible for detecting three attacks, packet ejection, hide and seek attacks and if a beacon is moved. From the information we received at the data gathering mode, we record three values per beacon, **tx-power**, **number of packets for time frame** and the **distance threshold**. Then we store in the detecting system.

Threshold based detection is useful when thresholds are clearly classifiable. However, BLE beacons RSSI values, which we use for our detection system, are heavily influenced by the environment. Consequently, these values have levels to noise. Finding static thresholds to monitor the changing in these values are usually hard to identify.

In order to overcome this problem, it is essential to complement the threshold detection engine with other mechanisms. For this purpose, we include this **machine learning engine** as part of our detecting system in order to detect attacks that couldn’t be detected by the threshold- based engine. This engine is responsible for detecting if a beacon is swapped with another beacon in the BLE infrastructure.
6.2.4 Location Model

In this model we attempt to measure the distance between the monitoring device and the moved or cloned beacons within the BLE infrastructure by using Received Signal Strength Indication RSSI, which is defined as the ratio of a received power to a reference power and its unit is decibel-milliwatts (dBm) as the following equation is dBm

\[ \text{RSSI} = 10 \log \left( \frac{P_r}{P_{Ref}} \right) \]

If there is a direct path between two nodes placed in an environment in which no signal interference occurs, the received signal power \( P_r \) and the distance \( d \) between the transmitting and receiving nodes can be calculated by using Friis transmission equation [H. T. Friis, 1946] as shown in the following

\[ P_r(d) = P_t \cdot G_t \cdot G_r \cdot \left( \frac{\lambda}{4\pi \cdot d} \right)^2 \]

Where \( P_r \) is the signal power at the receiver, \( P_t \) is the signal power output from the trimester, \( G_t \) is the gain of the transmitting antenna, \( G_r \) is the gain of the receiving antenna, \( \lambda \) is the wavelength, and \( d \) is the distance from the transmitter to the receiver. Based on Friis equation we can describes signal strength at distance \( d \) relative to some reference distance \( d_0 \) for which strength is known as showing in the following equation

\[ P_r(d) = P_t \cdot G_t \cdot G_r \cdot \left( \frac{\lambda}{4\pi \cdot d_0} \right)^2 \left( \frac{d_0}{d} \right)^2 = P_r(d_0) \cdot \left( \frac{d_0}{d} \right)^2 \]

Where \( P_r(d_0) \) is the received signal power at distance \( d_0 \), which is usually 1 meter for in-building environment, 100 meter or 1km for outdoor environment.
The Friis equation makes the assumption that propagation environment is essentially vacuous. However, in reality a stronger signal attenuation occurs when electromagnetic radiation becomes weaker as they pass through objects. To take into account stronger attenuation than only caused by distance, a larger exponent $\eta > 2$ is used and Friss equation can be changed as the following

$$P_r(d) = P_r(d_0) \cdot \left(\frac{1}{d}\right)^\eta$$

Where $\eta$ is the path-lose exponent which defines the rate at which the power fall. In free space the value of $\eta$ is equal to 2. However, in practice this value can vary depending on the surrounding environment. Table 6.1 [Mehra & Singh2013] shows the values of path loss exponent $\eta$ depends on the surrounding and the building type and can be calculated via the premeasurement [Liu and Wang 2012]

<table>
<thead>
<tr>
<th>Environment</th>
<th>Path Loss Exponent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Free space</td>
<td>2</td>
</tr>
<tr>
<td>Urban area cellular radio</td>
<td>2.7~3.5</td>
</tr>
<tr>
<td>In-building LOS</td>
<td>1.6~1.8</td>
</tr>
<tr>
<td>Obstructed in-building</td>
<td>4~6</td>
</tr>
<tr>
<td>Shadowed urban area cellular radio</td>
<td>3~5</td>
</tr>
</tbody>
</table>
Taking 10 times the logarithm of both sides on (4), the Equation (4) is transformed to the following equation

\[ 10 \log P_r(d) = 10 \log P_r(d_0) - 10 \eta \log d \]

10 \log P is the expiration of the power converted to dBm, which is the ratio of one power level, P, to one milliwatt reference power. The equation 4 can be written as the following

\[ P_{rd}(dBm) = P_{rd_0}(dBm) - 10 \eta \log d \]

Using previously mentioned RSSI equation in 1 with one milliwatt as reference power, equation 6 can be model as the following

\[ RSSI_d = RSSI_{d_0} - 10 \eta \log d \]

Where \(d\) is the distance between the receiver and transmitter, \(RSSI_d\) is received signal strength at the receiver, \(RSSI_{d_0}\) is the calibration signal strength measured at 1m, and \(\eta\) is the path loss exponent.

In order to calculate \(d\) in meters, equation 7 can be written as the following

\[ d = 10 \left( \frac{RSSI_{d_0} - RSSI_d}{10 \eta} \right) \]

And path loss exponent \(\eta\) can be calculated using the following equation

\[ \eta = \frac{RSSI_{d_0} - RSSI_d}{10 \log d} \]

By knowing the path loss exponent of the BLE infrastructure environment, the calibration signal strength at 1m, which is included within the BLE beacon broadcasted
massage, and the measured RSSI at the monitoring device, we can calculate the distance between the BLE infrastructure beacons and monitoring device.

To get the $\eta$ for our testing environment, RSSI values are measured at different distances from a BLE beacon, 0.5 m, 1 m, 2 m, 3 m, 4 m, 5 m, and 6 m respectively. At each distance 470 values are measured. The median values of the measurement and least square fitting are used to calculate $\eta$. Based on our calculation, the value of $\eta = 2.32$ is used for further measurements in our system.

In this section, we covered the architecture of our detecting system. In the following sections, we talk about the description of the tested attacks and the methods we used for detection.

6.3 Description of the BLE Tested Attacks

In this section, we describe four BLE attack scenarios that we produced to test our detection systems.

Beacon Swap Attack is a scenario assumes that the attacker has physical access to swap the location of two beacons or has no physical access yet is able to remotely change the configuration of two different beacons and swap their IDs.

Beacon Packet injection attack is a scenario assumes that the attacker has physical access to the BLE deployment area and has sniffed their IDs. Afterward, the attacker adds a beacon in a new location transmitting packets with the ID of the existing beacons.

Beacon physical attack is a scenario assumes that the attacker has a physical access to the BLE infrastructure and moved a beacon from one location to another.
Beacon hide and seek attack is a scenario assumes that the attacker has access to the deployment area and transmitting packets with the ID of the existing beacons but with different advertising transmitter power.

6.4 Beacon Swap Attack Detecting Method

Our proposed method includes two stages: offline profiling and online detection stage. In the offline stage the beacons profiles without attack is build. During this stage the monitoring device will collect RSSI samples from the installed beacons, filter them using moving median filter, label them with their beacon IDs and fed our machine learning model with the dataset to build the profile model. During the online stage the monitoring device scan the area of implementation for beacons frames. the RSSI values from each beacon will be collected and filtered using moving median filter to create datasets contain the filtered RSSI and beacon ID for each beacon discovered. The machine learning model will use the generated datasets to predict the beacon IDs for the datasets. If two beacons with swapped prediction are found, then that could be an indication that they are swapped in the implementation environment. Figure 6.2 shows an overview of the proposed beacon swap attack detection method.
6.4.1 Experiment Setup

In our experiment, we used the dataset collected during our normal operation profiling on moving median filter evaluation experiment (section 5.5) to train the classification algorithms running on our monitor device.

To simulate the swapping attack, we swapped the locations of beacon 1 and 2 and the locations of beacon 4 and 5. In order to evaluate our method, we did 300 tests across the day. Figure 6.3 shows the layout of testing area with swapped beacon and the location of our monitor device.
We used the following two metrics to evaluate the performance of our proposed approach:

- *Detection rate* which is the rate of the proportion of the test through which our approach detects the existence of the beacon swapping attack correctly.

- *False positive rate* which is the cases which our approach incorrectly determines the existence of the beacon swapping attack.
6.4.2 Experiment Results

Table 6.2 summarizes the results of our experiment. It shows that our proposed detection method can detect beacon swapping attack with 0 false positive rate. In addition, in our testbed our lowest detection rate is 0.973 using support vector machine classifier.

<table>
<thead>
<tr>
<th>Evaluation Metric</th>
<th>Swap 1 and 2</th>
<th>SVM</th>
<th>RF</th>
<th>DT</th>
<th>Swap 4 and 5</th>
<th>SVN</th>
<th>RF</th>
<th>DT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detection Rate</td>
<td></td>
<td>0.973</td>
<td>0.99</td>
<td>0.986</td>
<td></td>
<td>0.986</td>
<td>0.993</td>
<td>0.99</td>
</tr>
<tr>
<td>False Positive Rate</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

6.5 Beacon Packet Injection Attack Detecting Method

Our first approach was similar to the proposed framework in [Alotaibi et al., 2016]. The authors of that work used Random Forest in order to detect MAC address spoofing in wireless sensor networks. In their work, the authors labeled the RSSI values coming from the legitimate device 0 and the other values coming from 15 different locations in their testbed 1s. To distinguish a device with spoofed MAC address, they use Random Forest to predict if the packet comes from a legitimate device or not. They achieved 99.77% accuracy when the attacker is 8-13m. However, this accuracy dropped to 88% when the attacker is less than 4m away from the legitimate device. This approach has two limitation. First if the attacker is close to the legitimate device, this might lead the machine learning model to predict the attacker as a legitimate device. Secondly, an attacker can increase the signal strength in order to look closer to the legitimate device, which makes it hard to differentiate the RSSI values that come from the attacker. Due to these limitations, we adopted a different approach.
We found that during this type of attack the number of BLE packets broadcasted for a specific beacon ID increases since there is another beacon with the same ID operating in the BLE infrastructure. Based on this observation, during the data gathering mode we record the number of packets during a scanning window size, and we used it as threshold. During the monitoring mode if the number of packets for the same time frame is more than the one in the system, that could be an indication of a packet injection attack.

In order to measure the distance between the monitor device and the cloning device, we adopt the work presented in [Chen y 2010]. When a cloning device is present in the BLE infrastructure, the RSSI values from the legitimate beacon will be mixed with RSSI reading from the attacking device. Based on this, the RSSI values from the legitimate device will belong to the same cluster, while the RSSI values from the cloning device should form different cluster. Therefore, we conduct K-means clustering analysis on our monitoring device. The centroids returned by the K-means clustering can be used by our localization model to measure the distance between the monitor device and the cloning device.

6.5.1 Experiment Setup

Figure 6.4 shows the layout of testing area. The beacons transmission power is set to 0dBm and the advertisement interval set to 100ms. The monitor device scanned the environment for 5 second scanning window to measure the number of BLE frames received from each beacon before launching a packet injection attack. These values are stored in our threshold engine. The red dots in the figure represent the location where we launched our attacks. To simulate an attacker, we used a beacon device. The transmission power is set to 0dBm and the advertisement interval set to 100ms.
We did five rounds to attack the five legitimate beacons. In each round we configure the attacking beacon with the beacon ID of one of the beacons that we want to packet inject. We tested all five locations. In each location, we launched 100 attacks.

We used the following two metrics to evaluate the performance of our proposed approach:

- **Detection rate** which is the rate of the proportion of the test through which our approach detects the existence of the beacon packet injection attack correctly.

- **False positive rate** which is the cases which our approach incorrectly determines the existence of the beacon packet injection attack.
6.5.2 Experiment Results

Our detecting method was able to achieve 100% detection rate with 0% and detecting the attacking beacon even it was next to a legitimate beacon. Table 6.3 shows the actual distance and measured distance between the attacking beacon and monitoring device.

Table 6.3 The actual and calculated distance between the monitor device and cloned beacon

<table>
<thead>
<tr>
<th>Cloned Beacon ID</th>
<th>Beacon1</th>
<th>Beacon2</th>
<th>Beacon3</th>
<th>Beacon4</th>
<th>Beacon5</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Location</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Actual distance from monitor device</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
</tr>
<tr>
<td></td>
<td>Calculated distance from monitor device</td>
<td>0.52</td>
<td>0.92</td>
<td>1.1</td>
<td>1.2</td>
</tr>
<tr>
<td>2</td>
<td>Actual distance from monitor device</td>
<td>1.3</td>
<td>1.3</td>
<td>1.3</td>
<td>1.3</td>
</tr>
<tr>
<td></td>
<td>Calculated distance from monitor device</td>
<td>1.9</td>
<td>2.1</td>
<td>1.3</td>
<td>1.1</td>
</tr>
<tr>
<td>3</td>
<td>Actual distance from monitor device</td>
<td>3.1</td>
<td>3.1</td>
<td>3.1</td>
<td>3.1</td>
</tr>
<tr>
<td></td>
<td>Calculated distance from monitor device</td>
<td>3.3</td>
<td>2.8</td>
<td>2.7</td>
<td>2.4</td>
</tr>
<tr>
<td>4</td>
<td>Actual distance from monitor device</td>
<td>5.1</td>
<td>5.1</td>
<td>5.1</td>
<td>5.1</td>
</tr>
<tr>
<td></td>
<td>Calculated distance from monitor device</td>
<td>5.4</td>
<td>5.4</td>
<td>4.7</td>
<td>6.4</td>
</tr>
<tr>
<td>5</td>
<td>Actual distance from monitor device</td>
<td>8.5</td>
<td>8.5</td>
<td>8.5</td>
<td>8.5</td>
</tr>
<tr>
<td></td>
<td>Calculated distance from monitor device</td>
<td>12</td>
<td>12</td>
<td>8</td>
<td>10</td>
</tr>
</tbody>
</table>

The results show that our system has .6m error between the calculated and actual distance when the actual distance is between 0.5 and 5 m. However, this error increased to 3.5m when the actual distance is more than 8m.

6.5.3 Effect of Beacon Advertising Interval Evaluation.

The interval setting for a BLE beacon determines how often will transmit its advertising packet. This interval ranges from 100ms to 2000ms. However, Kontakt, one of the BLE beacon’s manufacturers, determined in their labs that higher interval settings over 700ms could cause major issues with signal stability.
Since the advertising Interval controls the number of advertising frames that a beacon can send in 1 second, this might affect the performance of our detecting methods. In this experiment we measure the detection rate of our method for two different scanning window sizes, 5 and 10 seconds. The goal of this experiment is to find the highest interval at which our system will have lowest detection rate.

In this experiment we used the same testbed in Figure 6.3. We configured beacon 5 with the same beacon ID of beacon 3 to simulate a packet injection attack on beacon 3. Beacon 5 was chosen because it is the farthest beacon from the monitor device in our testbed. We changed the advertising interval of attacker beacon until our monitoring device stopped detecting the packet injection attack.

We found that for scanning window size 5 seconds, our monitoring device stopped detecting any attacks with interval higher than 400ms. The detecting rate at 400ms is 75% for 100 attacked launched at that rate. However, with 10 seconds scanning window size, our monitoring device stopped detecting any attacks with interval higher than 1000ms. The detecting rate at 1000ms is 80% for 100 attacked launched at that rate. This was to be expected as when the size of the scanning window increases, the more attacking frames we can discover.

### 6.6 Beacon Moving Physical Attack Detecting Method

After applying moving median filter, we have lower signal fluctuation. During the data gathering stage, the monitor device receives the RSSI values from the legitimate beacon devices, filter them using moving median filter, calculating the distance using the localization model, and measure that upper and distance thresholds for each beacon. These values will be stored in the threshold-based detection engine. During monitoring mode, the
monitor device keeps monitoring the legitimate beacon devices distance. If the new calculated distance is out of the distance threshold, that could be an indication of a moved device.

6.6.1 Experiment Setup

In this experiment we used the same BLE infrastructure with 5 beacons installed. The beacons transmission power is set to 0dBm and the advertisement interval set to 100ms.

To launch an attack, we moved the installed beacons from their original location 10 time.

We used the following three metrics to evaluate the performance of our proposed approach:

- *Detection rate* which is the rate of the proportion of the test through which our approach detects the existence of the beacon moving physical attack correctly.

- *False positive rate* which is the cases which our approach incorrectly determines the existence of the beacon moving physical attack.

- *Distance sensitivity*. which is the distance that beacon should move before triggering our detecting system.

6.6.2 Experiment Results

Table 6.4 summaries the results of our experiment. It shows that our detecting system achieved 100% detection rate with 0% false positive rate. However, a beacon device has to move around 1.5m in order to trigger our system.
In our test, beacon 3 was more sensitive to the movement because it is installed closer to the monitor device, 0.5 m. On the other hand, beacon 5 installed 8 m away from the monitor device.

### 6.7 Beacon Hide and Seek Attack Detecting Method

During the data gathering mode, we record the tx-power broadcasted by each legitimate beacon in the BLE infrastructure in our detecting system. In order to detect a hide and seek attack we need to compare the received tx-power values in the monitoring mode to the one we recorded from the data gathering mode. If they are different, that is an induction of a cloned beacon broadcasting different tx-power.

#### 6.7.1 Experiment Setup and Results

In this experiment we used the same BLE infrastructure with 5 beacons installed. The beacons transmission power is set to 0dBm and the advertisement interval set to 100ms.

To lunch the hide and seek attack we used a Raspberry Pi3B device running a program written in python3.6 that allow us to broadcast beacon frames with different tx-power.

We placed our Raspberry pi within the testbed. Our system was able to detect the attacks with 100% detection rate and 0% false positive.
In this section we explained our detecting methods. In addition, the experiments that we produced to test our detection systems and their results.

During our experiments, we found the monitor device placement is significant to distinguish between the RSSI values received from the installed beacons with minimum overlapping. In the following section we explain the method that we used to help us install our monitor device.

6.8 Monitor Device Position Problem

Indoor positioning of monitoring devices within a Bluetooth beacon infrastructure has been a problematic phase for technical engineers in order to decide on how many monitoring devices are required and where these devices should be located. The locations and the quantity can be estimated. However, it isn’t always guaranteed that the estimation is one of the optimal alternative solutions.

Given the dimensions of an indoor space and the locations of the beacons installed there, we would like to add the minimum number of monitoring devices that would track the source of signal strength with no misleading overlapping.

To solve this problem efficiently, we created an algorithmic method to solve indoor positioning to improve tracking of beacons and distinguishing between the source of signal strength.

6.8.1 The Approach

To tackle indoor positioning of monitoring devices in a small room or office or in an open large sized public area, we deal with the area of interest as a 2-dimensional space and rely on Euclidean distance to estimate the signal strength.
Figure 6.5 Testbed room with 4 beacon installed

Figure 6.5 depicts a living room that has four beacons installed. The required input data we need to solve an indoor positioning problem is to get two type of data: the room dimensions (width/length) and the beacons’ positions as x and y axis values. Figure 6.6 shows the same living room in Figure 6.5 but converted into a 2-dimensional space of 5 by 4 meters with all beacons’ positions measured.
6.8.2 Algorithm Details

Once we have the required measurements about the space and the installed beacons, we can follow our algorithm. The algorithm decomposes the problem of indoor positioning into smaller ones, described in the following steps:

6.8.2.1 Build Up the Environment

The first step for the positioning algorithm is to initialize two data collections. The first collection is used to store the available beacons we have in a room with their (x,y) coordinates. The other data collection is used to store all possible positions or points within the room space with their (x,y) coordinate values. These points are simply defined by scanning the room and incrementing x axis and y axis one at a time by one unit (e.g meter). Then, we calculate the Euclidean distance between points and beacons. We store these distance values in a global collection keyed by point number and valued by point’s distances to beacons.

E.g:
6.8.2.2 Create Cliques of Non-overlapped Signals

Once we calculated the distances to beacons for every point, we can decide which beacons are overlapped for a point. We rely on distance to define beacons’ signals overlapping. If two beacons have the same distance to a specific point, these two beacons are overlapped for that point. We set a threshold value to accept the difference (e.g. 1.5 meter) in distance between all pair of beacons to every point. Then, we create a graph of non-overlapped beacons for every point. Two beacons are related (have an edge) if there is no overlap between their distances to the point. Graph nodes represent beacons and edges represent no-overlap relationships between beacons. Using an undirected graph data structure, using NetworkX, helps us to create all maximal cliques. Maximal cliques are the largest complete subgraph containing a given node. We also maintain a value of the minimum difference of distances calculated for all pair of beacons. This minimum difference is the weight value assigned to a point, such that a point with high weight would most likely guarantee to prevent any overlapping between beacons signals. Another value we consider the most to evaluate points is the maximum clique size. The largest clique size the better for non-overlapping in beacons’ signal. Here’s is an example of the steps to create a clique list for point P12, depicted as a graph in Figure 6.7

P12’s distance to beacons:
[['B4', 2.23606797749979], ('B1', 1.0), ('B2', 3.0413812651491097), ('B3', 3.3541019662496847)]
P12's clique list: [['B1', 'B4', 'B3'], ['B1', 'B2']]
P12's weight: 1.118033988749895
P12's max clique size: 3

Figure 6.7 Beacon Clique Relation for Point 12

The following Algorithm shows the pseudocode for creating a clique list for all points.

```
Algorithm 1: Create Point's Weight and Clique List

Inputs:
pointDistanceOfBeaconList

Initialize:
G ← newGraph()
threshold ← 1.1
weight ← 100

for (b_i, d_i) ∈ pointDistanceOfBeaconList do
  for (b_j, d_j) ∈ pointDistanceOfBeaconList do
    if b_i ≠ b_j then
      overlapDistance ← |d_i - d_j|
      if overlapDistance ≥ threshold then
        G e edge(b_i, b_j)
        if weight > overlapDistance then
          weight ← overlapDistance
        end
      end
  end
end

cliques ← findClique(G)
```

6.8.2.3 Create All Possible Positioning Solutions

After calculating a clique list for each point with its weight, we can get all possible minimal combinations of points that can track beacons’ signals confidently with no overlaps. Then we can find the best solution that has the smallest number of points with a
high weight. Algorithm 2 shows the pseudocode for creating all possible positioning solutions within a specified limited space area.

```
Algorithm 2: Create All Possible Solutions of Non-overlapping Point Sets

Inputs:
- beaconList
- pointWithWeightList
- pointWithCliqueList

Initialize:
- solutionList ← ∅

for \((p_x, w_x) \in \text{pointWithWeightList}\) do
  for \(c_x \in \text{pointWithCliqueList}[p_x]\) do
    weight ← \(w_x\)
    tempSolution ← \(c_x\)
    if tempSolution = beaconList then
      solutionList ← tempSolution
      break
    else
      for \((p_y, w_y) \in \text{pointWithWeightList}\) do
        for \(c_y \in \text{pointWithCliqueList}[p_y]\) do
          if \(c_y \neq \text{tempSolution}\) then
            tempSolution ← \(c_y\)
            if \(w_y > \text{weight}\) then
              weight ← \(w_y\)
          end
        end
        if tempSolution = beaconList then
          solutionList ← tempSolution
          weight ← \(w_x\)
          tempSolution ← \(c_x\)
        end
      end
    end
  end
end
```

By applying algorithm 2 to the environment depicted in Figure 6.6, we get the most desirable solutions depicted in Figure 6.8, that shows P5 then P16 as the best solutions of one monitor device that can differentiate between the 4 beacons, then a list of P21 and P29 as the next best solution of two monitor devices. A device at point P21 to monitor 2 beacons and norther device at point P29 to monitor the other two beacons.
Finally, Figure 6.9 shows the position of a monitoring device selected for the given environment.

In order to test our algorithm, we picked a random location within the testbed room to test against the location picked by our algorithm. The goal of the test is to find if the given
location by our algorithm would give better prediction if machine learning is used to predict the beacon ID of the surrounding beacons.

Our experiment involves two stages: the offline stage and the online stage. In the offline stage, the legitimate device profile is built. During profiling, the monitoring device scans the environment for a scanning window size of 5 seconds. During this window, RSSI values from each beacon will be collected and labeled with their beacon IDs. We use the dataset to train the classifier and store the model for predicting new packets. In the online stage, the monitor device will scan the environment for BLE frames for a scanning window size. During this window size, the RSSI values from each beacon will be collected to create datasets for each beacon discovered. These datasets will be fed to the machine learning model. The model then predicts if the packet comes from the same beacon ID or not. We sued SVM as our classifier. The classifier has 0.85 F1score using the random location we picked. On the other hand, this score is improved to 0.93 using the location given by our algorithm. This result shows that our algorithm can be used to help finding the best fit location for our monitor device.

6.9. Conclusion

In this chapter we proposed a novel RSS-based detection system. We provide an overview of the architecture and a description of the system’s building blocks. In addition, we describe the attacks that could be detected by our system, the detection methods for each attack. Our system has 100% detection rate detecting beacon physical moving attacks, hide and seek attack, and packet injection attack. In addition, it has 99 detection rate
detecting beacon swapping attacks. These results outperformed the results in the work presented in the literature review section 2.2.

In addition, in this chapter we described the algorithm that we used in order to position our monitor device in the testbed.

Even though our proposed system could provide good results. However, we find out from our experiments that it is very sensitive for major changes in the testbed. For example, changing the furniture around the testbed, requires running new data gathering and training for both machine learning engine and threshold engine. In addition, combining packet injection attack with hide and seek attack, would make it harder to discover the location of the attacking device. Therefore, based on those limitation in next chapter we propose an enhanced beacon protocol (eBeacon) to provide new unclonable and authenticated advertainment.
Chapter 7

Enhanced BEACON Protocol

7.1 Introduction

Bluetooth Low Energy (BLE) Beacons introduced a novel technology to enhance services with location and context awareness by constantly broadcasting a static unique identifier in order to advertise their presence for other observer devices. However, due to the design and implementation of the current BLE beacon protocols, BLE beacon devices are typically vulnerable to different types of attacks related to device cloning and spoofing.

While the Bluetooth Specification contains some security mechanisms, which mostly aimed to protect the data transmitted between paired devices, the only security technique provided by the standard specification for devices that operate in the broadcaster mode is the use of random MAC address [Blue2019]. Yet, this does not protect against BLE beacons spoofing attacks because the attacker could utilize the existence of the broadcasted static unique identifier inside the advertised payload to clone the beacon. In other words, such attack can be done because the beacon advertisement packets are open and static and therefore can be easily recorded and cloned to a different beacon device.

In attempt to solve this problem we propose an enhanced beacon protocol (eBeacon) to provide new unclonable and authenticated advertainment. By using the Manufacture Specific Data profile in the Bluetooth Core Specification [Blco2019] , we modified the BLE advertisement packet to contain a timestamp and Advanced Encryption Standard Cipher-based Message Authentication Code (AES-CMAC) tag [RFC4493] that could only be linked back to the beacon by authorized devices.
The contributions of our work are as follow:

- Proposal of a new beacon protocol.
- Identification and formulation of individual components constituting the new protocol.
- Present its implementation.
- Proposal of future beacon protocol for Bluetooth5.

The rest of the chapter is organized as follows: the design of the eBeacon and its components are presented and discussed in depth in Section 2. The implantation of the protocol is presented in section 3. In section 4, the design of future eBeaconV5 is presented and discussed, and finally the chapter is concluded with summary and discussion in section 5.

7.2 Proposed eBeacon Protocol

In this section, the design and components of the proposed eBeacon architecture for securing a BLE beacon advertising is presented. The proposed architecture is composed of three main components: 1) eBeacon Beacon devices, that will be used to broadcast our new beacon protocol; 2) eBeacon verifier that will verify the authenticity and integrity of the broadcasted protocol; and 3) eBeacon advertising Protocol. The roles and the formulations describing the characteristics of the components will be presented below.

7.2.1 Design Principles and Decisions

Before we discuss our proposed protocol, we outline the main design principles and decisions we made in the system.

A. The proposed protocol has to follow the Bluetooth Core Specification.
B. Beacon devices usually rely on batteries to maintain their operational status.

Therefore, the proposed protocol has to adopt a simple cryptographic scheme.
AES is chosen in our protocol since it is already used as part of the Bluetooth Specification [Blue2019]

C. The proposed protocol has to fit in a single advertisements packet in order to minimize the number of the broadcasted packets.

D. Public key encryption-based solution was not chosen because it requires long message but BLE broadcast PDU can hold up to 31 bytes including 4 bytes header.

E. Due to the broadcasting nature of BLE beacons, the proposed protocol should rely on unidirectional advertisements only.

F. The eBeacon resolver should be able to verify the authenticity of the eBeacon Beacon devices and the broadcasted packet integrity.

### 7.2.2 eBeacon Design

In our protocol, rather than broadcasting a static advertising packet, we modified the advertisement packet to include a timestamp and AES-CMAC tag that change every second. Figure 7.1 shows the design of the eBeacon packet.

![Figure 7.1 eBeacon packet](image)

The operation of the proposed architecture is as follow:

- A prior to development each eBeacon beacon has a 14byte symmetric key (K) that is shared and known by the eBeacon verifier. In addition, 2bytes unique identification number (ID_\text{b}) and the measured signal strength at a distance of 1 meter to the beacon (TX-power) are set.
- The eBeacon beacon computes 128 bits key MAC_{\text{key}} on the concatenation of ID_\text{b} and K.
• The beacon uses the $\text{MAC}_{\text{key}}$ to compute an AES-CMAC on the concatenation of BeaconID, the timestamp and Tx-Power on padding and generate a 128 bits CMAC tag.

\[
\text{CMAC}_{\text{tag}} = \text{AES-CMAC}_{\text{MACkey}}\{\text{BeaconID}||\text{timestamp}|| \text{Tx-Power}\}
\]

• The identifying information in eBeacon advertisements consists of the beacon ID, timestamp, CMAC$_{\text{tag}}$ and TX-power.

• Whenever an eBeacon verifier observes an eBeacon advertisement, it checks the advertisement timestamp. The advertisement is accepted if the received timestamp is not more than some threshold $(d)$. Subsequently, it computes its own version of the CMAC$_{\text{tag}}$ and compares it with the received CMAC$_{\text{tag}}$. This way the verifier can identify the beacon and check the integrity of received advertisement.

• Any phone/device that supports the eBeacon protocol when observes an eBeacon advertisement can transfer the observed data to the eBeacon verifier that will identify the beacon, check the advertisement packet integrity and notify the phone/device that observed the beacon. Figure 7.2 shows the operation of the proposed architecture.

Figure 7.2 eBeacon operation
7.3 Implementation and Experimental Setup

We implement the eBeacon beacons and verifier using Raspberry Pi 3 model B which has 1.2quad-core ARM Cortex A53 CPU, 1 GB LPDDR2-900 SDRAM with Bluetooth Low Energy 4.0. support and runs Raspbian Stretch operating system.

For programming the Bluetooth Low Energy protocol, python 3.6.3, Linux Bluetooth protocol stack Bluez, pybluez and pygattlib libraries are used. In addition, Pycryptodome Python package is used for cryptographic programming.

During implementation for debugging and analyzing the transmitted eBeacon protocol, ViewTool Hollong Bluetooth 4.0/4.1/4.2 BLE Sniffer/Analyzer is used. Hollong was chosen over other sniffers in the market because it can scan and capture all three (37, 38, 39) advertising channels concurrently.

During initialization 3 Raspberry Pi devises were programed as eBeacon beacons and assigned ID, secret key, and the calibrated TX-power and placed the beacons at 3 chosen locations in 5 by 7m room.

For the verifier role, another Raspberry Pi3 was configured with the three beacons IDs that has to verify and the shared key.

7.3.1 Example BlueZ packet for eBeacon Packet.

This part contains information on how to decode an eBeacon advertisement packet. For example, we have the following packet.

04 3E 2B 02 01 00 00 C9 4D 7E EB 27 B8 1F 02 01 06 1B FF 4C 00 18 1E 00 01 01 AD B2 ED 9E 1C CB 33 7C A8 02 65 60 B8 7D 48 2A 17 1C B8 C2

This packet can be broken down as follows:

04: Packet type HCI_EVENT_PKT
3E: Event LE_META_EVENT
2B: Packet length = 43 This is the length of the LE meta event packet.
02: The meta event is EVT_LE_ADVERTISING_REPORT
01: The number of advertising reports in the packet = 1
00: BT device address type
C9 4D 7E EB 27 B8: BT device address (MAC address) = B8:27:EB:7E:4D:C9
1F: Advertising data length = 31

The key part of that packet is the Bluetooth Advertisement, which can be broken down as follow:

02: The number of bytes that follow in the first AD structure.
01: Flags AD type
06: Flags value 0x06 = 00000110
   bit 0 (OFF) LE Limited Discoverable Mode
   bit 1 (ON) LE General Discoverable Mode
   bit 2 (ON) BR/EDR Not Supported
   bit 3 (OFF) Simultaneous LE and BR/EDR to Same Device Capable
   (controller)
   bit 4 (OFF) Simultaneous LE and BR/EDR to Same Device Capable (Host)
1B: number of bytes that follow in the second AD structure
FF: Manufacture specific data AD type
4C 00: Company identifier code (0x004c == Apple) this code can be changed to any company code that would support eBeacon protocol.
18 1E: eBeacon protocol type
00 01: Beacon ID
01 01 AD B2: timestamp
ED 9E 1C CB 33 7C A8 02 65 60 B8 7D 48 2A 17 1C: AES-CMAC tag
B8: The 2’s complement of the calibrated TX-power
C2: The measured RSSI by the device.

7.3.2 System evaluation

In our test we want to evaluate the functionality of the system in term of broadcasting the designed eBeacon packet and checking the integrity of received advertisement on the verifier device. In addition, measure the time required to create the message M, compute CMAC_tag, send eBeacon packet, and checking the integrity of the received eBeacon packet.

The configured system successfully broadcasted and verified the received eBeacon packets. In addition, the verifier device was able to detect when and old packet is received
and when the content of the advertisement is changed. For creating a new message, the system takes 0.0008 second. In addition, it takes 0.001 second to compute $\text{CMAC}_{\text{tag}}$ and 0.08 second to broadcast eBeacon advertisement packet. On the other hand, it takes the verifier device 0.09 second to verify the integrity of the received eBeacon packet.

### 7.4 Security Analysis

In order to clone our protocol, the attacker should be able to generate a valid new AES-CMAC tag or reuse an old AES-CMAC tag. In this section, we discuss the security features embedded in the design of our protocol eBeacon.

#### 7.4.1 Security of CMAC

In our proposed protocol, the output of the CMAC verification determines the assurance that on the receiver of the message side if the output is INVALID, then the message is definitely is not authentic. On the other hand, if the output is VALID, then the message is authentic and in result it was not corrupted in transit. However, In the second case, a party without access to the secret key or the CMAC generation process may try to guess the correct CMAC for the message. Tough, if the attacker selects a CMAC at a random set of strings of length $\text{Taglen}$ bits, the probability is $1$ in $2^{\text{Taglen}}$ that the CMAC will be valid. Therefore, larger values of the CMAC tag length provide greater protection against such an event. In our protocol we use 128 bits as output for our CMAC tag.

In addition, an attacker might try to brute-force the secret key space in order to discover the used secret key in the system. If the attacker can determine the CMAC key, then it is possible to generate a valid CMAC tag for any input values in our packet. For
example, in our protocol with a key size of 128 bits, if the attacker has one known text-tag pair then the attacker can compute 128 bits CMAC tag on the known text for all possible keys. At least one key would produce the correct tag and that key can be used by the attacker for future attacks. However, this phase of the attack takes a level of effort proportional to $2^{128}$ since we use 128 bits key size, which would take $7.66 \times 10^{25}$ years to crack [Seagate2008].

Furthermore, an attacker may be able to exploit a collision to produce a valid CMAC for a new message. If the attacker can find two messages $M_1$, $M_2$ with $\text{CMAC}_k(M_1) = \text{CMAC}_k(M_2)$ then for any string $N$, we have $\text{CMAC}_k(M_1||N) = \text{CMAC}_k(M_2||N)$. Hence, if the attacker can find such a collision, the attacker can ask for the CMAC of $M_1||N$ and know the CMAC of $M_2||N$. However, a collision is expected to exist among a set of 264 messages for AES-CMAC [NIST2005]. In addition, based on National Institute of Standards and Technology (NIST) recommendation, the CMAC keys should be limited to no more than $2^{48}$ messages. Within this limit, the probability that a collision will occur is less than one in a billion for AES-CMAC [T. Iwata, 2003].

**7.4.2 Reply Attack on the Protocol**

**Setup:** eBeacon protocol is broadcasted at a location $l$ with AES-CMAC tag. An attacker at the same location listens to the medium and records the broadcasted packet at time $t$. The attacker does not have knowledge of the secret key. The advertised AES-CMAC tag is only valid during a time $d$. At time $t+d$, eBeacon beacon calculates a new AES-CMAC and broadcast new advertisement.
**Attack:** At time \( t + d + \delta \), where \( \delta > 0 \), the attacker broadcasts the recorded eBeacon advertisement with AES-CMAC tag at the same location \( l \) so a legitimate user can listen to it.

**Response of the system:** eBeacon verifier will listen to the broadcasted message and check the timestamp. If the timestamp is greater than \( t + d \) then the message is invalid.

Our proposed eBeacon protocol can be used as a countermeasure to beacon devices spoofing, packet injections, and hide and seek attacks. However, can’t protect against piggybacking attacks, where the attacker or third-party developers build an application on top of a Bluetooth Low Energy infrastructure that they don’t own. In other words, because detecting beacon IDs is possible, an attacker can include those IDs in his/her program. For example, a store A installed number of beacons to advertise its products within the store by using program A. Then an attacker already knows store A’s beacons IDs can make a program B that advertise store B by using store A infrastructure. When a customer installed program, B pass a store A, his/her device shows events or discounts related to store B.

Such an attack is possible because the beacon ID is scannable, open, static, and can be mapped to a specific location. Therefore, as a countermeasure we utilize Bluetooth 5.0 core specification to propose eBeacon5 Protocol, which can protect against both beacon spoofing and piggybacking attacks.

**7.5 Proposed eBeacon5 Protocol.**

When it comes to develop a secure BLE beacon protocol, one of the most challenging issues to deal with is BLE beacons only allocate a small chunk to customize their
advertising PDU since it is built on top of Bluetooth 4.0 Core specification and can only
hold up a payload of to 31 bytes. However, adopted on December 2016 as the latest revision
of Bluetooth Core specification, Bluetooth 5 sets the stage for the next generation of
Bluetooth products with higher speed, longer range, and increased advertising capacity.

Bluetooth 5 allows advertising packets to be up to 255 bytes long. This accomplished
through using the Extended Advertising feature that is introduced in Bluetooth
Specification Version 5.0. Bluetooth5 defines two types of advertising channels, Primary
Advertising Channel and Secondary Advertising Channel. New field called AuxPtr is
transmitted as part of the advertisement header on channels, 37, 38, 39, which are knows
as the Primary Channels in the context of Bluetooth5 advertising. The AuxPtr includes the
secondary channel number that advertising payload will be transmitted on, which is one of
the other channels in the 0-36 channel number range previously only used in Bluetooth 4
for transmitting data for connection events.

In our eBeacon5 protocol design we utilize the Extended Advertising feature to
transmit a larger payload that supports Authenticated Encryption with Associated Data
(AEAD) encryption form [RFC5116] to encrypt the broadcasted beacon ID and check the
authenticity of the advertising PDU in order to protect against spoofing and piggybacking
attacks.

### 7.5.1 Authenticated Encryption with Associated Data (AEAD)

Many cryptographic applications require both confidentiality and message
authentication services to ensure that the data is only available to those authorized to obtain
it and to ensure that the data has not been altered by unauthorized entities. Many applications use an encryption method and MAC together to provide both confidentiality and message authentication with each algorithm using an independent key. However, Authenticated Encryption with Associated Data (AEAD) can provide both services using a single crypto algorithm.

The authenticated encryption operation has four inputs, as following:

- A secret key K, which could be a size of 128, 192, 256 bit depends on the used AEAD algorithm.
- A nonce N, A 96-bit nonce that shouldn’t be reused for the same key.
- An arbitrary length plaintext P, which contains the data to be encrypted and authenticated.
- Arbitrary length additional associated data AAD, additional data that should be authenticated with the key but is not encrypted.

The output from the AEAD is a single output that has two parts appended:

- A ciphertext C, which is at least as long as the plaintext
- 16 byte authentication tag T, which is the output of the authentication function used running on C, AAD, and N.

For the authenticated decryption operation four inputs are used, K, N, AAD, and C as defined above. the AEAD encryption function is run on the ciphertext, producing the plaintext. In addition, the AEAD authentication function is applied to the ciphertext, AAD, and nonce to calculate T. the calculated tag is compared to received tag and the message is authenticated if and only if the tags match. The authenticated decryption operation will return FAIL whenever the inputs N, C and AAD were crafted by unauthorized adversary that does not know the secret
7.5.2 eBeacon5 Protocol Design

eBeacon5 Protocol is a new beacon protocol that could be broadcasted by BLE beacons that support Bluetooth v5 and can be checked by BLE verifiers for authenticity.

Figure 7.3 shows the content of eBeacon5 as follows:

1. Two bytes company identifier code, and two bytes eBeacon5 protocol type so the verifier could distinguish between eBeacon5 advertisement and another Bluetooth PDUs.
2. 32 bytes ID reference tag (IRT). A beacon that supports eBeacon5 will compute this tag by running HMAC function on its 4 bytes beaconID (ID_b) to convert the ID to an identifier for its associated encryption information in the verifier device. This value will be changed every predefined interval.
3. 12 bytes nonce (N), this will be used as part of AEAD computation. This value will be changed every time the AEAD tag is computed.
4. 6 bytes timestamp that is used as a countermeasure against replay attacks.
5. 24 bytes AEAD tag. The beacon ID will be encrypted and for AAD, IRT, the timestamp and TX-power will be included.
6. 1 byte TX-power, the 2’s complement of measured signal strength at a distance of 1 meter to the beacon.

<table>
<thead>
<tr>
<th>CompanyID (2 bytes)</th>
<th>AD Type (3 bytes)</th>
<th>IRT (32 byte)</th>
<th>N (12 byte)</th>
<th>TimeStamp (6 byte)</th>
<th>AEAD (24 bytes)</th>
<th>TX Power (1 byte)</th>
</tr>
</thead>
</table>

Figure 7.3 eBeacon5 packet

7.5.3 eBeacon5 Process

- A prior to development each beacon has 32 bytes symmetric key (IRT_key), that will be used to compute IRT and another key AEAD_key that will be used to generate the AEAD tag. These keys will be shared and known by the verifier. In addition, 4bytes ID_b and TX-power are set.
- The beacon generates IRT
  \[
  \text{IRT} = \text{HCMA}_{\text{IRTkey}}(\text{ID}_b)
  \]
• The verifier generates IRT for every beacon in the environment and store in a
database that includes the beacon ID, IRTkey, and AEADkey.
• The beacon generates N and use it with AEADkey to compute AEAD tag, where
the ID_b will be encrypted and AAD = {IRT|| timestamp || TX-power}.

\[ \text{AEAD}_{\text{tag}} = \text{AEAD}_{\text{key}}(\text{ID}_b, \text{AAD}) \]

• The beacon will broadcast eBeacon5 advertising PDU and generate new N.

\[ \text{eBeacon5}_{\text{adv}} = \{\text{IRT, N, timestamp, AEAD}_{\text{tag}}, \text{TX-power}\} \]

• Whenever an eBeacon5 verifier observes an eBeacon5 advertisement, it checks
the advertisement timestamp. The advertisement is accepted if the received
timestamp is not more than some threshold (d). The verifier will use the IRT in
order to retrieve the beacon associated encryption information from its local
database, which includes the actual beacon ID, and keys. Subsequently, it
computes its own version of the AEAD_{\text{tag}} and compares it with the received
AEAD_{\text{tag}}. This way the verifier can identify the beacon and check the integrity of
received advertisement.
• After a configurable period, the beacon devices and the verifier will generate new
IRT.

Notice that in our new proposed eBeacon5 protocol the actual beacon ID is not
broadcasted within the advertisement packet. However, we run HMAC function on the
beacon ID every configurable period to create a changeable ID, IRT, which is included
within the advertisement packet instead. Consequently, eBeacon5 can be used to protect
against spoofing, packet injections, and hide and seek attacks in addition to piggybacking
attacks since the beacon ID is not static anymore.
7.6 Conclusion

In this chapter we proposed a new beacon protocol based on BLE v4 that can be used as a mitigation against spoofing, packet injections, and hide and seek attacks. In addition, we identify and formulate the components of the new protocol, preset its implantations. Finally, we proposed a furfure beacon protocol for Bluetooth5.
Chapter 8

Integrate e-BLE-BEACONS INTO R2D2 System

Data security attracts a lot of attention recently due to the huge impact of recent cyber-attacks on major government and private organizations. For example, the data breach at U.S. Office of Personnel Management (OPM) results in the loss of personnel data for 4.2 Million individuals. 21.5M Social Security Numbers were stolen from the related background investigation database, among them 19.7M from applicants, 1.8M from their spouses, and 1.1 M fingerprints [OPM 2015]. Yahoo reported on December 14, 2016 that in August 2013 unauthorized third party stole data associated with more than one billion user accounts based on forged cookies without passwords. Even though encryption may prevent the data breach in OPM incident, it does not address that of Yahoo account loss. New data protection techniques are needed to enhance data security.

8.1 Secure Right Path Right Place Data Delivery

Data security can be enhanced by not only strong encryption techniques but also by restricting who can receive the data, as well as where and how they are received. In this chapter, we focus on how to specify and verify where and how the data are received. The challenging issue here is how to utilize trustworthy 3rd party verification mechanisms to check whether a requester is in the designated area specified by the creator and whether the delivery follows a desired path.
Figure. 8.1 shows the proposed system components for realizing R2D2 service. The Document Authoring System consists of web based tool for creating key, encrypting data/files, specifying the geolocation area, and uploading the key, encrypted data, geolocation constraints, and email distribution list to the Document Distribution System. Emails will be sent to the designated recipients by the Document Distribution Server. The user follows the instruction in email to visit the designated “Right Place” area and click the link for data. When the Document Distribution Server receives the request, it either requests the trustworthy location verification system(s) such as Wifi or Bluetooth systems to confirm the presence of the user in specified area, or use value-added SDN system to consults with the involved router along the designated “Right Path” to see if they have observed the client request packets. The requests from wrong place or wrong path will be rejected. The requests from right place or right path will be accepted.
Since we are considering the attackers or adversaries could generate the requests for the secure or sensitive documents, we cannot trust the location provided by the receiver. We need to verify, through trustworthy 3rd party or subsystems under our control, where the requests or receivers located and to check over what type of paths the secure data will be delivered. Note that the environments where R2D2 services can provides data security do not limit to the classified environment. They can include the private sectors too for sensitive or proprietary data protection or regulation compliance. The logging mechanism provided by R2D2 systems can be used for satisfying the auditing required by regulations in certain applications.
R2D2 opens new research areas on the design of the language for specifying the path and location constrains, and the techniques for verification whether the constrains are satisfied given a specific R2D2 to a certain recipient.

To verify the requester is in the right place, we investigate and conduct experiments to see if our campus Wi-Fi system can be used as 3rd party trustworthy system to confirm the location of a mobile user. Here we assume the mobile user has a registered account with the campus and their mac address show up in signal records kept by the campus Aruba system. We have serious discussion with the campus IT since there are concerns about the privacy issue in terms of accessing the signal records and tracking users. A script was written with the interface to the Aruba system to retrieve the signal records for specific users. The Document Distribution System can request signal records of a specific user by providing credentials on secure connection.

In our experiment, we want to check the following:

- If the connected access points that reported by the mobile client matches the ones recorded by Aruba system and compare the signal strength collected from a mobile host and Aruba system at a given time and location.
- The maximum number of access point reported by the mobile host and compared it to Aruba system at a given time and location.
- The minimum number of access point reported by the mobile host and compared it to Aruba system at a given time and location.

**8.2 R2D2 Experimental Testbed**

Our testbed located on in a campus building with two floors. Figure 8.2 shows seven Aruba access points located on the first floor and nine access points on the second floor.
that run both 2.4 and 5 GHZ radio. Our mobile host carried by the user being MacBook Pro running OS X EL Capitan with wireless card that is IEEE 802.11 a/b/g compatible.

Figure 8.2 The building with wifi base stations (greendots)

Our testbed environment would be classified as being open along the hallways and closed elsewhere. The access points provide overlapping coverage in portions of the building and together cover the entire 2-storey building.

Figure 8.3 shows the RSSI valued collected from the mobile client and Aruba system for the 17 Wi-Fi AP and 29 different locations within the building. There is 5.90 dBm average difference between the collected RSSI information from the remote host and Aruba System with $\sigma = 3.95$ dBm. In addition, the maximum number of the surrounding AP discovered by the mobile client is 12 and the minimum number is 7 access points.
Figure 8.3 RSSI values collected from the mobile client and Aruba system.

The Aruba system has accuracy about five meters. In addition, the system needs two minutes in order to update the user Wi-Fi information in the system. Finally, our initial approach was to use the RSSI values on the client side to locate the user and compare it to the RSSI values reported from the system to cross validate the location. However, localization information may be blocked in order to protect the user privacy. For example, Apple prevents applications from getting a list of Wi-Fi access points around the user and their RSSI values as a result. It is only possible to get the SSID and BSSID of the network that the user is currently connected to. However, the application cannot read and report the client Wi-Fi MAC address.

Therefore, the system limitation, the variation in the RSSI values, and manufacture company restrictions could be issues and should be taken in consideration before adopting Wi-Fi as a 3rd party system to validate the user location.
8.3 Bluetooth Low Energy Verification System

In this section, we describe the design and component of the proposed BLE system that could be used to verify the requester is in the right place.

8.3.1 How to define a room using BLE

Location verification schemes can be categorized as on-spot and in-region verification [Wei 2013]. In on-spot verification, a system discovers the absolute location of a user. However, in the in-region verification the user claims to be in a particular region, which could be a room, a building, or other physical area, and the system accepts or rejects the claim. Our proposed system is an example of in-region verification.

This technique needs to fix a number of proximity beacons in order to define a Secure Region (SR). This region represents the location where the user has to be at to grant access to the secure data. In this region “r” is the maximum distance that the user is allowed to be from a region’s beacon. A region match occurs when a user is in a region that is included in the SR. Therefore, in Figure 8.4 U1 location matches, however U2 location doesn’t match.
In order to define our secure region and “r”, we place the beacon at the center of the room and measure the RSSIs at different reference points starting with 1 meter away from the beacon and increment another meter, and so on until we cover the entire room. In addition, we measure the RSSIs value outside the room when the room door is close and open. In order to convert the RSSIs values into distance the path lose model [Kumar 2009] will be used.

However, The RSSI values at mobile client flautists over time because of several noise factors. Therefore, various filters will be compared to smooth the RSSI values before using. For example, average filter can be calculated by collection a few packets from each reference node within a second and calculate the average RSSI for that reference node.
After calculating the distance at every reference node and the near proximity region with its “r” value can be defined for the room. For more than one beacon the same steps will be used to estimate the distance from each beacon and then Weighted Centroid Localization Algorithm (WCL) [Quanda 2014] to estimate the position of the unknown node. Figure 8.5 shows the flow of the positioning method.

![Figure 8.5 BLE Positioning Method.](image)

### 8.4 BLE Verification System

Our proposed BLE verification system has the following components:

- **A mobile application**: this App will be used to authenticate the user to the system and collect the user location information for verification.

- **Witness BLE beacons**: These fixed beacons will be used to define the secure region, broadcast their beacon IDs and RSSI values that will be collected by the mobile application.

- **Notary devices**: Those are Raspberry Pi 3 model B with built-in Wi-Fi, 10/100Mbps LAN card, and Bluetooth Low Energy support. The notary will be used to monitor the installed Witness BLE beacons, report if a physical attack
occurs, and to collect and send the BLE information in the room to R2D2 system to verify the claimed locations by the users as in Figure 8.6.

Figure 8.6 BLE Room Layout

- **An Authentication and location verification server**: It has the following modules: First, Authentication Module, which will be used to authenticate the users, generate OTP that will be used as a shared password between the server and the mobile application, and random device ID that will be used by the mobile application. Context Validation Module is the second module in the system. This module will collect the location information from the mobile application and Notary devices in order to check if the user is within the predefined secure region. Access Policy Verification module is responsible of checking the user contexts against the predefined policy, that includes time, location, and date. Finally, Token Generator Module that will be used to generate random tokens to be used to prove the user location to The Document Distribution System.
Figure 8.7 shows an example of the message flow between the entities of our proposed system.

1. The user uses a mobile phone, which act as a client device to detect the BLE network. When the user enters the proximity region associated with beacons, the application will trigger the user to login to authentication and location verification server providing the username and password.

2. Once the authentication is successful, the server will send a random phone ID and OTP that will be used to encrypt the later massages send between the mobile device and the server.

3. The mobile device will collect the IDs and RSSI values of the surrounding Witness beacons in order to build the BLE context information.

4. The application will use the OTP to encrypt the BLE context information in addition to the identity context, which include the username and phoneID, and time
and send this information to the Context validation module. The context validation module will decrypt the information using the same OTP and store the information.

5. Next the device will start broadcast its BLE information using the random generated phoneID.

6. This information will be detected by the Notary devices and will send it to the server including the other RSSI and ID values seen.

7. On the server, the BLE information submitted by the client will be compared against the information by the BLE mentoring network and the location will be calculated.

8. If the two calculations confirmed that the user is inside the secure region, within a distance not more than “r” for that region, then user will be checked against the predefined policy.

9. The Access Policy Verification module will check the user information time, date, and location against the policy.

10. Finally, if validated, the user will get one time token that the user mobile application will send to The Document Distribution System to prove the user location before decrypting the encrypted file. The Document Distribution System can request the token records from the authentication and location verification server on secure connection.

In our proposed system, we will use BLE. However, different technologies that use RSSI for localization could be added. For example, we can use the university access points as witnesses and the Wi-Fi management system as Notaries to validate the claimed location. Figure 8.8 shows a generic location verification system where BLE notary, Wi-
Fi notary, Cellular, GPS location system can be integrated to facilitate the delivery of R2D2 services.

![Figure 8.8 Generic Location Verification System.](image)

### 8.5 Right Path Verification Challenge

To verify whether the request and delivery packets follow certain path within the network, it requires the routers along the path to timestamp the packets and reports back to the Document Distribution System. Not only it is time-consuming, but also the function is not readily available for most switches. However, with the emerging SDN networks, it is possible to design a value-added layer around the existing standard switch and prototype the timestamp service or a service for verifying the flow for getting more confidence. In [OTMEN2018] we present the architecture and implementation of OTMEN: a monitoring system that offloads customized traffic monitoring tasks from the SDN controller to agents...
located along with edge switches in data center. The OTMEN can be extended to operate on the campus network for right path verification. Figure 8.9 shows an overview of the OTMEN architecture. The agents in OTMEN system only preform local monitoring for active flows and ports statistics in the co-located virtual switch based on requests received from OTMEN controller. The OTMEN controller basically extends the traditional SDN control plane with four additional modules:

**Monitoring delegator**: enables SDN applications to define periodical monitoring tasks, which will be delegated to agent(s) based on the specified switch ID and switch-port number(s) in the monitoring request. When the agent(s) return the monitoring results, the monitoring delegator makes these data available for SDN applications. Also, the monitoring results can be saved in a permanent database when it is required. Here the data can be used to detect whether the path has been redirected or rerouted with different delay and path characteristics.

**Alerting manager**: enables operators to set event triggers. For example, when a flow volume exceeds a certain threshold or deviated from the normal range, the agent will send an alert2 notifying the controller about this flow, so the controller can take action as early as possible.

**Agent manager**: handles the messages between the modules in the controller and OTMEN agents. Besides, it manages OTMEN agents, i.e., allows operators to disconnect and shutdown the agents on remote servers.
**OTMEN APIs**: represent the unified interface for OTMEN modules in the controller and enable SDN applications to configure various monitoring tasks.

Figure 8.9 An overview of OTMEN architectural components.

### 8.6 Potential Related Services

With the capabilities developed, various services can be provided. This includes:

**Secure-E911 campus service.** When the authority or family has emergency to locate the persons within the campus or organization. Note that GPS will not work inside the building. The Wi-Fi and BLE type systems however can be used to locate a person with a building. In this case, the privacy preserving mechanism to be developed will be overwritten.

**Privacy Preserving Group Location Service.** A group can specify where and when the group members will allow to be located.
8.7 Conclusion

In this chapter, we have presented a R2D2 concept for enhancing the data security by ensuring their delivery to the right place over the right path. Software tools are also developed to allow a campus Wi-Fi system to be queried for signal strengths of a mobile user observed by the base stations. Finally, a design of BLE verification system is proposed in order to integrate and provide R2D2 services.
Chapter 9

Lessons Learned

Lessons Learned Working with BLE Beacon Devices

One of the first lesson I’ve learned, they are not reliable and have to be tested before implementation. During one of the experiments that I conducted, one of the beacons that I installed was giving high advertising rate even when I changed the advertising rate value. The solution to that problem was to change its battery. This leads to lesson number two. Always check the battery level of all beacons before implantation and change it if there is a big difference between the levels. Lesson number three. Always do your own calibration, the tx-power value broadcasted by the beacon might not reflect the actual value you should get at 1m and might lead to inaccurate distance estimation. Finally mount the beacons 3-5 meters above the ground.

Lessons Learned Working with BLE Signals

They need to be filtered or smoothed. Lesson number two they are sensitive to the changes in the environment during the day. In other word, what works in the morning might give different results at night. Therefore, to get an accurate measurement of a system dealing with BLE signal, those measurement have to be collected at a different time of the day. It takes time but help.
Lesson learned Working with Raspberry Pi Devices.

Raspberry Pi devices use SD cards as a storage which might have reliability issue. I lost a month long worth of work because one of the Raspberry Pi SD cards became very seriously corrupt. Therefore, frequent backup is a mandatory. In addition, selecting the position of the monitor device should not be done randomly. Rather there should be a systematic algorithm to follow that guarantees the minimum overlaps between receiving signals.
Chapter 10

Future Directions

In the future the range of machine learning algorithm can be extended to included other techniques such as deep learning.

During our research we covered different attack scenarios. However, there are other attacks that we didn’t covered yet. For example, in [Brauer S 2016] the authors present a selective jamming attack on Bluetooth Low Energy advertising. Such an attack could corrupt all BLE advertisement beacon frames in result this could make the BLE beacon source undiscoverable. Therefore, part of the future direction is to investigate alternative attack scenarios and counter measurements.

On January 2019 The Bluetooth Special Interest Group (BSIG) published Bluetooth Core Specification V5.1 including more enhancement and added features to Bluetooth Low Energy (BLE) technology. One of these enhancements is Direction Finding. The current Bluetooth proximity solutions use signal strength to estimate distance. However, with the new direction-finding features, it is possible for Bluetooth devices to determine the direction of a Bluetooth signal transmission. This new feature offers two different methods for determining the angel that a Bluetooth signal is being transmitted from. The two methods are Angel of Arrival (AoA) and Angel of Departure (AoD).
Each of the techniques requires one of the two communicating devices to have an array of multiple antennae as shown in Figure 10.1. So as a future research direction is to include this feature in our detecting system to enhance localizing the attacking devices.

Another new feature released by BSIG is Bluetooth mesh network. One of the features that BLE lacked since the beginning was the capability of supporting many to many topologies where multiple BLE devices can send each other messages and relay messages to other devices. Another future direction is to study the security of the new BLE mesh technology and how it could be utilized in asset tracking, beacon reporting and real time location services and the privacy issues that might occur.
Chapter 11

Conclusion

In this research, we have designed and developed a new the Bluetooth protocol for authenticating devices conforming to Bluetooth version 5.0 standard. Machine learning based defense techniques for Bluetooth Beacon-based geolocation System are developed and analyzed. It was enhanced with filtering techniques for reducing noise impact. The vulnerabilities of the current BLE beacon systems were analyzed and a mobile attack application for exploiting such a vulnerability was developed. Integrating the new BLE protocol and BLE Beacon Defense techniques into the secure data delivery to right location system was also presented.
Bibliography


[BLE Core Specification], BLE Core Specification Volume 6, Part B, section 4.4.2.12


[NIST2005] Special Publication 800-38B


Workshops (PerCom Workshops), 2016 IEEE International Conference on. IEEE, IEEE, Sydney, NSW, Australia, 1–6.


Appendix A – INSTALLATION and CONFIGURATION

This appendix is included as a separate file.
http://cs.uccs.edu/~gsc/pub/phd/eashary/doc/ReadMe.pdf