EXPLORING BLUETOOTH FOR RECEIVED SIGNAL STRENGTH INDICATOR-BASED SECRET KEY EXTRACTION

by

Prarthana Lakshmane Gowda

A thesis submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Master of Science

in

Computer Science

School of Computing

The University of Utah

December 2012

The University of Utah Graduate School

STATEMENT OF THESIS APPROVAL

The thesis of **Prarthana Lakshmane Gowda**

has been approved by the following supervisory committee members:

Sneha Kumar Kasera	, Chair	**09/25/2012** Date Approved
Neal Patwari	, Member	**07/05/2012** Date Approved
Robert Ricci	, Member	**07/05/2012** Date Approved

and by **Alan Davis**, Chair of

the Department of **School of Computing**

and by Charles A. Wight, Dean of The Graduate School.

ABSTRACT

Current approaches to secret key extraction using Received Signal Strength Indicator (RSSI) measurements mainly use the WiFi interface. However, in the presence of jamming adversaries and other interfering devices, the efficiency of RSSI-based secret key extraction using WiFi degrades and sometimes the key extraction may even fail completely. A possible method to overcome this problem is to collect RSSI measurements using the Bluetooth interface. Bluetooth appears to be very promising for secret key extraction since the adaptive frequency hopping technique in Bluetooth automatically detects and avoids the use of bad or interfering channels. In order to collect Bluetooth RSSI values, we design a protocol where Alice and Bob use Google Nexus one phones to exchange L2CAP packets and then we measure the RSSI for each received packet. We use a prequantization interpolation step to reduce the probability of bit mismatches that are caused due to the inabililty to measure the time-duplex channel simultaneously by Alice and Bob. We then use the ASBG quantization scheme followed by information reconciliation and privacy amplification to extract the secret key bits. We conduct numerous experiments to evaluate the efficiency of Bluetooth for secret key extraction under two different mobile environments - hallways and outdoors. The secret bit rates obtained from these experiments highlight that outdoor settings are better suited for key extraction using Bluetooth when compared to hallway settings. Furthermore, we show that for very small distances such as 2 ft, the number of consecutive "0" RSSI values and bit mismatch is too high to extract any secret key bits under hallway settings. Finally, we also show that Bluetooth key extraction in outdoors achieves secret bit rates that are comparable to WiFi, even when using lower transmit power than WiFi.

For my family and friends.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to take this opportunity to gratefully acknowledge a number of people who have constantly motivated and supported me in making this thesis possible. First and foremost, I would like to express my gratitude towards my advisor, Professor Sneha Kumar Kasera, for being the guiding force in my research endeavors at this university. He has been a constant source of motivation and has always shown faith in me. I am grateful to Professor Neal Patwari, for his active involvement and valuable guidance and suggestions. I would like to thank Professor Robert Ricci for being on my committee and for taking the time to provide quality guidance and feedback. I am indebted to Sriram N. Premnath for his time and active involvement in my research right from the beginning. I would also like to thank Saurav Muralidharan, Arijit Banerjee, Suchit Maindola, Shobhit Gupta, Axel Y. Rivera and Srikanth Manikarnike for making the long research hours in the lab enjoyable. My sincere thanks to Ann Carlstrom and Karen Feinauer for being such wonderful graduate advisors. I am grateful to my parents, Yashoda and Lakshmane Gowda, who have always put my interests and well-being ahead of theirs. Finally, and most importantly, I would like to thank God for always giving me the strength, courage and perseverance to help me achieve my goals.

CHAPTER 1

INTRODUCTION

Secret keys are fundamental for any private communication. They are used for authentication, integrity, and confidentiality. Traditional systems use either public key or symmetric key cryptosystems for this purpose. The security of public key mechanisms relies on the difficulty of factoring large numbers. Though these techniques are being widely used, they are susceptible to brute-force attacks using modern hardware like GPUs [19]. Hence, there is a need for alternate methods to establish secret keys which do not rely on factoring of integers.

More importantly, concerns about the security of public keys has given way to research in the field of quantum cryptography in order to establish keys that do not rely on the assumption that factorization of large integers is computationally infeasible. Although there has been some promising development in this regard [5, 6], quantum cryptography based methods are expensive and hence, are not used commonly. Therefore, there is a need for other efficient and practical techniques for establishing secret keys.

Under wireless settings the reciprocity and location-specific characteristics of radio wave propagation can be exploited to generate secret keys. Reciprocity of radio wave propagation means that the multipath properties of the radio channel, which include the gains, phase shifts, and delays, are identical on both directions of the link within the coherence time. These multipath characteristics of a wireless link vary over time due to the change in environment, which could be caused by movement of objects or people in the surroundings. Also, the multipath characteristics are location specific, i.e., the variations measured by two parties of a link are specific to their location and any other eavesdropper who is separated by a few wavelengths cannot measure the same multipath characteristics [18].

Recent work shows that in wireless environments, the received signal strength can be used as an indicator of the multipath characteristics of a wireless link [13, 11] and hence, can act as a source of secret keys. Moreover, RSSI-based secret keys can act as one time pad by constantly generating the stream of secret bits, hence, providing information theoretic

security as opposed to factorization of integers in public keys.

1.1 Motivation

Current approaches to secret key extraction using RSSI measurements mainly use the WiFi interface [11]. However, there has not been much work which focuses on other readily available interfaces for key extraction. Other interfaces which facilitate same-channel communication between two devices can be used for RSSI-based secret key extraction. Bluetooth is a commonly used interface and is readily available in existing devices. Furthermore, Bluetooth operates at a low power compared to WiFi, which is an important factor in battery-limited devices like smartphones. However, there is no existing work on secret key extraction using Bluetooth. Additionally, Bluetooth also offers some jamming resistance. Although Bluetooth operates in the 2.4GHz band, it automatically detects channels which are busy and hops over a number of other free channels in a random manner. As a result, unlike WiFi, Bluetooth is less susceptible to interference in WLAN and similar environments. Even though jamming is not evaluated in this thesis, a standard that is robust against jamming and is easily available is an attractive option for exploration.

1.2 Thesis contributions

In this thesis:

- We explore the use of Bluetooth wireless interface for robust, low-power, and efficient secret key extraction based on RSSI. We use a Bluetooth mode that is inherently less susceptible to interference.
- We design a simple protocol to sample the wireless channel for collecting RSSI measurements.
- We implement the key extraction mechanism for Bluetooth on smart phones and evaluate this mechanism for various hallway and outdoor settings.

1.3 Thesis overview

The remainder of this thesis is organized as follows: In Chapter 2 we discuss the background and necessary Bluetooth concepts. Following this, we present our adversary model in Chapter 3. Chapter 4 discusses the basic RSSI-based key extraction mechanism. We discuss our design and key extraction methodology for Bluetooth in Chapter 5. Implementation details are presented in Chapter 6. We describe the experimental setup and evaluate our

results in Chapter 7. Next, we discuss the related work in Chapter 8. Finally, we present our conclusions in Chapter 9.

CHAPTER 2

BLUETOOTH BACKGROUND

Bluetooth is a wireless technology that operates in the unlicensed 2.4 GHz band. It uses frequency hopping spread spectrum technology [1], in which signals are transmitted by rapidly switching or hopping frequency channels. In the basic frequency hopping mode, Bluetooth divides the 2.4 GHz band into 79 channels and hops 1600 times per second in a pseudo-random fashion. This random sequence (also called the hopping sequence) is known to all nodes in a Personal Area Network (PAN). In this chapter, we first discuss the details of Bluetooth frequency hopping and later explain the various layers of the Bluetooth protocol stack followed by the Bluetooth device power classes.

2.1 Adaptive frequency hopping

Apart from the basic frequency hopping mode, Bluetooth version 1.2 also incorporates an adaptive frequency hopping mode. The technique of adaptive frequency hopping (AFH) [9] makes Bluetooth an ideal interface to extract secret keys in the presence of WiFi jamming adversaries or during heavy network traffic. AFH allows Bluetooth to adapt to the environment by identifying fixed sources of interference and excluding them from the list of hopping channels. So, if there is a wireless device interfering with the Bluetooth device, then Bluetooth identifies those channels used by the interfering device as bad channels and remaps the hopping sequence to exclude them. However, the hopping sequence should include a minimum of 20 channels out of a total of 79 available channels [2]. The $AFH\_channel\_map$ indicates the channels that are unused and the ones that are occupied by other interfering devices. This parameter contains 79 one-bit fields where each bit indicates the status of the corresponding channel. A bit 1 in the nth position indicates that the nth channel is used and 0 indicates a channel that is not used by other devices.

The AFH_map is one of the inputs to the algorithm which calculates the hopping frequencies. Before the hopping frequencies are obtained, a hop sequence generator is used to select a pseudo-random sequence of numbers which is later used to generate hopping

frequencies [1]. Figure 2.1 shows the diagram of the various parameters used for calculating the hopping sequence. K-offset is a number that is initialized to 24, and changes by either 8 or 24 depending on the mode. N is a counter that is initially set to 0. The sequence selection input specifies the type of hop selection to be done; in the AFH mode it is set to *Adapted channel*. The hop sequence generator does several operations on the input parameters such as addition, XOR, and permutation operations to generate a random sequence [1]. The output of the sequence generator [2] is a pseudo-random sequence of channel indices which has to be mapped on to actual frequencies for hopping. The equation that maps a channel index to a channel frequency is as below.

$f = 2402 + k \text{ MHz}, k = 0, ..., 78.$

Note that the Radio Frequency (RF) channels are spaced 1 MHz apart.

AFH mode also requires that the two communicating parties over Bluetooth, one master and the other slave, use the same channel for communication. This same-channel communication mode reduces the frequency hopping rate by half. Hence with the AFH mode ON, the Bluetooth channel hops 800 times per second.

Figure 2.2 illustrates basic frequency hopping versus adaptive frequency hopping in the presence of WiFi interference. Note here that the AFH specifically avoids the interfering WiFi channel.

To summarize, Bluetooth appears to be very promising for secret key extraction mainly

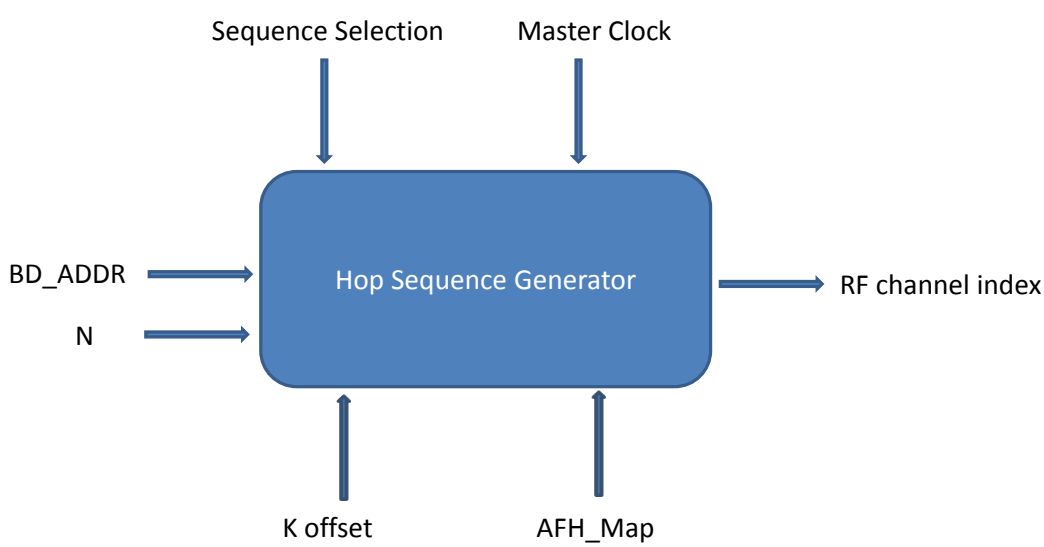

Figure 2.1. Hop sequence generator

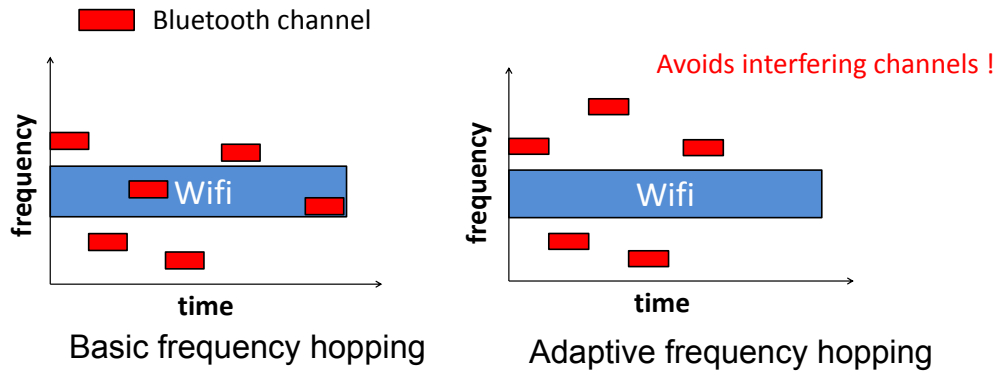

Figure 2.2. Basic hopping frequency versus adaptive hopping in the presence of interference

due to the following two reasons:

- It automatically detects and avoids the use of bad or interfering channels.
- Since it is same-channel communication, we can still exploit the reciprocity property of radio waves to extract secret bits as in WiFi.

2.2 Bluetooth protocol stack

Bluetooth has a layered protocol architecture. Figure 2.3 shows the layered architecture of the Bluetooth stack. The protocol stack is split into two parts:

- Controller stack
- Host stack

The controller stack is implemented on the hardware and includes the Bluetooth radio and a microprocessor. The host stack deals with high level data and is implemented as software that can be installed on top of the operating system. The controller provides protocols for connection establishment and also to manage link parameters while the host stack deals with protocols for packet handling and to set parameters for service discovery, Bluetooth profiles, etc. [14].

2.2.1 Bluetooth radio

This is the hardware device that is responsible for transmitting and receiving electrical signals in the ISM band. Each device should have specific transmitter and receiver characteristics that are defined by certain standard bodies, in order to maintain compatibility.

2.2.2 Base band layer

This is the physical layer that handles synchronous and asynchronous connection-oriented links. It also interacts with and supports the link layer in managing certain link characteristics like power control, inquiry, paging etc. The handling of these link characteristics at the hardware level affects secret key extraction because:

- It does not give the user the freedom to adjust the power level.
- It does not give the user much control over the rate at which the raw RSSI values can be collected. We further explain this in Section 5.1.1

2.2.3 Link manager

The link manager includes the link manager protocol, which is responsible for setting up the link and controlling the link characteristics. With the help of link messages, the link manager on one device communicates with the link manager of the device on the other end of the link in order to control certain link characteristics such as transmit power, QoS, level of security etc. The link manager protocol measures the received signal strength of the

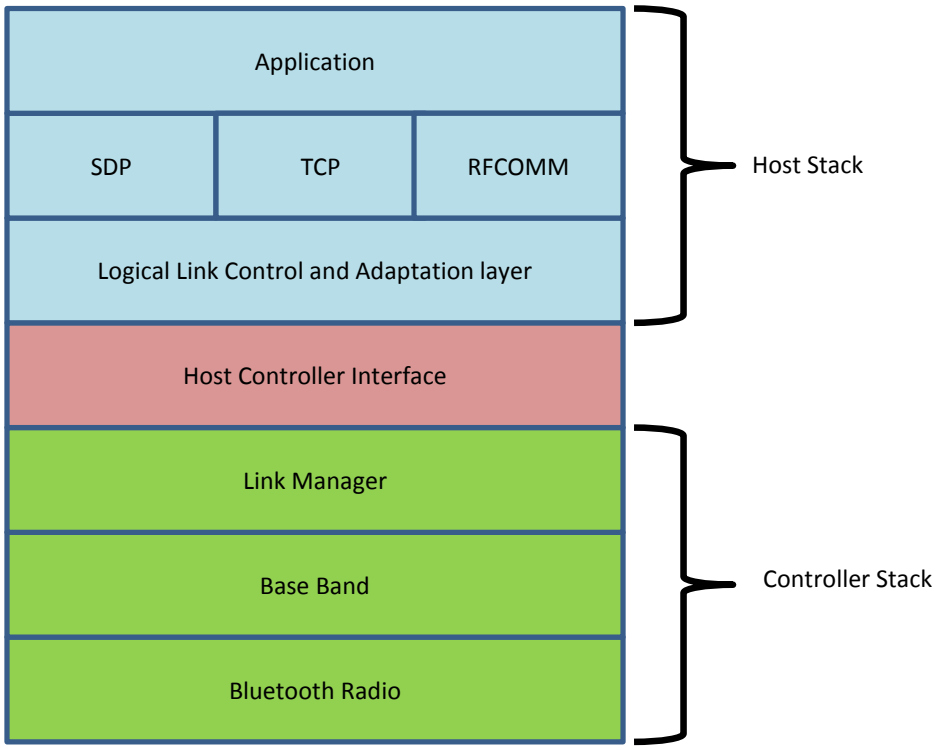

Figure 2.3. The Bluetooth protocol stack

received packets and exchanges control messages with other link managers on corresponding Bluetooth devices. Based on these received control messages the Bluetooth devices increase or decrease their transmit power. In other words, the *user* of the Bluetooth device does not have any explicit control over the transmit power of the device.

2.2.4 Host controller interface

The Host Controller Interface (HCI) layer is an interface between the controller stack and the host stack of Bluetooth. In the BlueZ protocol stack [3], the HCI is a software interface that provides several commands to interact with the baseband and the link layer. The HCI includes commands to read status and control registers of the link layer and also commands to test the functionality of Bluetooth hardware.

2.2.5 Logical link control and adaption protocol

The Logical Link Control and Adaption Protocol (L2CAP) is built over the base band layer and is responsible for providing asynchronous connectionless or synchronous connection-oriented services to upper layers. Some of its other functions include multiplexing of data between higher layer protocols, segmentation and reassembly, quality of service management, etc. [14, 2]. Since L2CAP is closer to the HCI and link layer, and also since it interacts with them directly, in our work we exchange L2CAP packets to measure the RSSI.

2.2.6 RFCOMM, SDP, TCP

The Radio Frequency Communication Protocol (RFCOMM), Service Discovery Protocol (SDP), and Telephony Control Protocol (TCP) are a set of protocols that are built on top of L2CAP. These protocols are responsible for providing various services that are used by Bluetooth applications. For example, RFCOMM is used for serial port emulation, SDP defines several Bluetooth profiles, and TCP is used for intercom and cordless telephony.

Among the many layers of the Bluetooth protocol stack, the layers that are of importance to secret key extraction are:

- Link layer - since it can record the RSSI value of every received packet.
- The Host Controller Interface layer - since it provides a command interface to access link layer registers and read the RSSI values.

- L2CAP layer - since it can help in exchanging L2CAP packets to obtain the RSSI values.

2.3 Bluetooth power classes

Based on power, Bluetooth transmitters are classified into three power classes [14]. The classes along with their transmit power and operation range are listed in Table 2.1.

The Bluetooth specification requires class one devices to compulsorily implement power control; however, it is optional for devices belonging to power class two and three. Most of the class two and class three devices today, do not come with a power control feature [14].

The Google Nexus One smartphones that we use in our work belong to the category of power class 2 with a maximum transmit power of 3dBm [4].

Table 2.1. Maximum transmit power for different classes of Bluetooth devices.

Device Class	Maximum transmit power (mW)
1	100
2	2.5
3	1

CHAPTER 3

ADVERSARY MODEL

In our set up we assume that the adversary, Eve, can sample and measure the channel between herself, and Alice and Bob at the same time as them. However, she cannot be positioned very close to either Alice or Bob. Specifically, Eve should be several wavelengths (of the radio waves being used) away from both of them. Otherwise, she can measure the correlated channel between Alice and Bob and can obtain a similar bit stream extracted by Alice or Bob [11, 18] . The algorithms for key extraction along with the parameters used are public. We assume that the data integrity is protected, i.e., the adversary is not interested in manipulating the messages between Alice and Bob. The adversary has the power to affect the communication channel between Alice and Bob by moving objects randomly in the path of key extraction. However, she cannot control the movements to an extent as to significantly increase the coherence time of the channel.

The algorithm for calculating the AFH hopping sequence [2] is a publicly known standardized algorithm. Also, the AFH hopping sequence is common to all the nodes in the piconet. The AFH sequence is broadcast over the air and an adversary outside the piconet can listen to it. However, we note that, although AFH does not add any security to the system, it is considerably difficult to jam Bluetooth due to the hopping mechanism. In our adversary model, all the nodes in the piconet are genuine and we also assume that the adversary does not try to jam the Bluetooth interface. Our scheme does not involve authentication of the genuine parties and is not immune to active person-in-the-middle attacks. However, since we use a connection-oriented protocol for exchanging packets any authentication used by Bluetooth during connection setup is still valid.

CHAPTER 4

KEY EXTRACTION METHODOLOGY

In this chapter we discuss a general methodology for RSSI-based secret key extraction. Most RSSI-based key extraction methodologies include several stages. Firstly, the two wireless nodes Alice and Bob need to sample the wireless channel to obtain their individual set of RSSI values. These RSSI values are integers and must be quantized to extract binary bits. This process is called quantization [18]. After this step, both Alice and Bob would have established an initial bit stream. However, this stream of bits cannot be used as the final secret key since there may be bit mismatches and those mismatches must be corrected. This process of error detection and correction is done in the information reconciliation stage [7]. To establish a strong secret key, we must ensure that an adversary should not be able to predict any part of the secret key by having information about the key extraction process. Privacy amplification [10] stage is employed in this regard to obtain the final secret key bits. Figure 4.1 depicts the steps involved in RSSI-based secret key extraction. In the sections that follow, we discuss each of these steps in detail.

4.1 Quantization of RSSI samples

Quantization is the process of constraining a relatively large or continuous set of values (integer RSSI) to a relatively small discrete set (bits). In our set-up, Alice and Bob collect a time series of measurements of variations in the wireless channel by exchanging probe packets. This series is then quantized individually by both the parties, using the following thresholding mechanism to obtain an initial bit sequence. For quantization of RSSI values, we use the Adaptive Secret Bit Generation (ASBG) scheme proposed by Premnath et al. [18, 11] This quantizer provides very high entropy while maintaining a high output secret bit rate.

In the ASBG scheme, Alice and Bob divide the consecutive RSSI measurements into blocks of appropriate size - which is a configurable parameter. For each block, they independently calculate two thresholds, upper threshold q^+, and lower threshold q^-, where

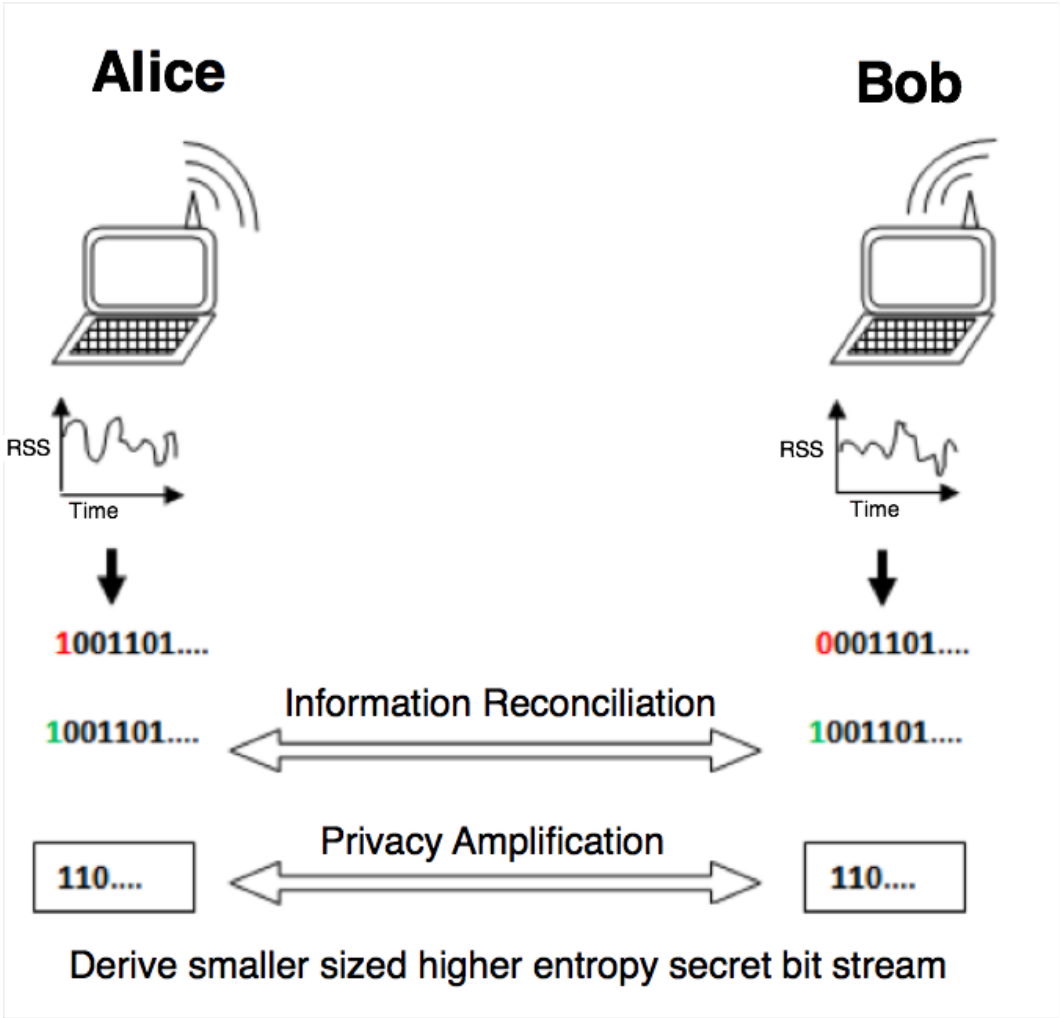

Figure 4.1. RSSI-based secret key extraction

$q^+ = \mu + \alpha \times \sigma$ and $q^- = \mu - \alpha \times \sigma$ [11]. Both Alice and Bob parse their respective RSSI values, and any value lesser than the lower threshold is encoded as zero, any value greater than the upper threshold is encoded as bit one, and all values that lie between the upper and the lower threshold are discarded. Alice and Bob maintain a list containing indices of discarded RSSI values and exchange it with each other so that they exclude all such indices from further consideration for secret key extraction.

Furthermore, in our work, we also consider a method to effectively reduce the effects of shadow-fading on the bit stream. Shadow-fading is caused due to the obstructions from large objects in the environment. We must reduce the effects of shadow-fading so that the bit stream obtained after quantization is mostly from the hard-to-predict effects of small

scale fading as opposed to the predictable effects of shadow-fading. Note that small scale fading is caused because of relative motion between the radios and different objects in the environment. Premnath et al. [17] have designed an analytical scheme to reduce the effects of shadow-fading. Their results show that choosing the right block size for the running average window in the quantization scheme reduces the effects of shadow-fading. Their analysis shows that the right block size depends on the relative speed of the nodes and the sampling rate. We use this technique to calculate the right block size for quantization of Bluetooth RSSI values.

Thus, at the end of the quantization step, Alice and Bob would have individually established an initial key bit stream which is then used as input for the information reconciliation stage, to remove any bit mismatches.

4.2 Information reconciliation of quantized bits

After Alice and Bob have individually generated the initial bit sequence by encoding the RSSI values using the quantizer, they must now check for correctness of their keys. Under ideal conditions Alice and Bob should measure the channel at the exact same time and hence, they should not have any asymmetry in their bit streams. However, due to the half-duplex mode of operation of the transceivers, they have to measure the channel one direction at a time. This, combined with other factors such as presence of noise and interference, hardware limitations, manufacturing variations, vendor-specific differences etc. create asymmetry in the bit stream generated by Alice and Bob. Alice and Bob must ensure that both of them obtain the same sequence of bits by revealing minimal information, while communicating on an insecure public channel. This procedure is called information reconciliation [7].

As in the case of Premnath et al. [18], we use cascade-based information reconciliation technique [7] for Bluetooth interfaces, in our work as well. Cascade is an iterative protocol where either Alice or Bob randomly permute their bit stream, divide this permuted stream into blocks of appropriate size, and calculate the parity of each block. This parity information as well as permutation information is communicated to the other party on a public channel. The other party uses this information and repeats the same permutation and parity calculation steps on their bit stream. For such blocks whose parity does not match, a binary search is performed to find a small number of bits that can be changed to make the block match the parity information. These steps are repeated until the probability of success reaches a desired value. Cascade is a probabilistic protocol and we can achieve high

probability of success by selecting the right number of passes and the optimal block size.

4.3 Privacy amplification of reconciled bit stream

Ideally, Alice and Bob should probe the channel only once within the coherence time of the channel. However, since the coherence time depends on the movements in the environment, which is unpredictable, calculating the exact coherence time is difficult. This implies that Alice and Bob could have probed the channel more than once within the coherence time. Therefore, the bit stream can exhibit a short-term correlation between subsequent bits. Moreover, during the information reconciliation stage, some information about the bit stream is leaked into the insecure public channel. We need a mechanism to minimize the correlation and also remove the leaked information from the final bit stream. The privacy amplification [10] step solves these problems by using 2-universal hash functions; $h : \{1...M\} \rightarrow \{1..m\}$ where $M > m$, m is chosen based on the entropy of the bit stream and also on the amount of information leaked during information reconciliation [11].

CHAPTER 5

KEY EXTRACTION IN BLUETOOTH

As we have described in the previous chapter, after Alice and Bob measure the RSSI value of the channel by sending probes to each other, they execute the following three steps to extract the shared secret key: quantization, information reconciliation and privacy amplification. Most of the work on secret key extraction involves some or all of these three stages. Premnath et al. [18, 11] used an adaptive lossy quantizer in conjunction with Cascade-based information reconciliation [7] and privacy amplification [10] in their secret key extraction work with WiFi using laptops. In our research we use their method for Bluetooth interface on smartphones. However, even before we apply quantization for Bluetooth RSSI values, we have to consider the method of sampling the channel itself. Furthermore, in our work, we use an interpolation step [17, 15] to reduce the probability of mismatch in the measured RSSI values between Alice and Bob. Figure 5.1 describes the steps used in RSSI-based secret key extraction for Bluetooth.

In the following sections we describe the sampling and interpolation stages in detail. The rest of the steps remain the same as described in Chapter 4.

5.1 Sampling the Bluetooth channel

Bluetooth facilitates two methods to obtain RSSI values [16]:

- Inquiry-based RSSI: This method measures the RSSI of inquiry packets in inquiry mode during device discovery.

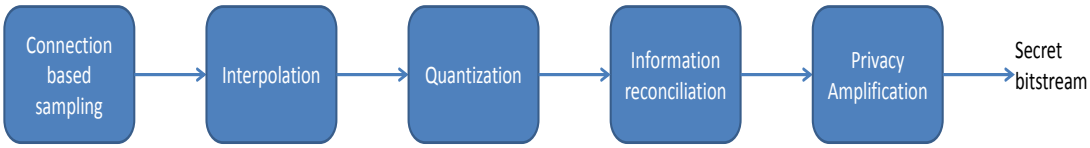

Figure 5.1. RSSI-based secret key extraction for Bluetooth

- Connection-based RSSI: This method measures RSSI of data packets after a connection is established.

In our thesis, we have evaluated both these methods and in this section we discuss them with their pros and cons.

5.1.1 Inquiry-based RSSI

A Bluetooth device enters into inquiry mode in order to discover other devices. During the device discovery state, the master node (device doing the inquiry) and the slave node (device that wants to be discovered) enter several substates as shown in Figure 5.2.

The details of operation of the various substates are as follows.

5.1.1.1 Inquiry substate

When a Bluetooth device wants to discover other devices in its range, it enters into the inquiry substate. In this state, the device doing the discovery repeatedly broadcasts an inquiry message by sending out inquiry packets called ID packets, over different frequencies [1].

During this time, the device doing the inquiry follows a separate hop sequence called the inquiry hopping sequence. This hop sequence uses 32 frequencies equally distributed over 79 MHz. The 32 frequencies are divided into 2 hop patterns of 16 frequencies each. In this mode the device hops much slower when compared to the adapted frequency hopping mode and waits for response packets from any device. In order to collect adequate response packets, an inquiry substate lasts for at least 10.4 seconds [2]. A device can also be put in a periodic inquiry mode. In this mode, the period between two successive inquiry messages is determined randomly for each device, in order to minimize collisions between two inquiring devices. We can obtain the raw RSSI value of an inquiry-response packet received by the master by putting the device in the "inquiry with RSSI" mode.

5.1.1.2 Inquiry scan substate

Any device that wants to respond to an inquiry message should be in the inquiry scan substate. In this state, the responding device waits for inquiry message packets by hopping onto 16 frequencies. In this state, the device uses the inquiry scan hopping sequence [2]. This hopping sequence is similar to the inquiry hopping sequence in the sense that it covers 32 response frequencies that are in a one-to-one correspondence to the current inquiry hopping sequence. In order to completely scan for the 16 inquiry frequencies, the Bluetooth

specification defines the scan period to be 11.25 milliseconds by default and less than or equal to 2.56 seconds [2].

5.1.1.3 Inquiry response substate

When a device in the inquiry scan substate receives an inquiry message, it enters the inquiry response substate. In this state, the device sends back an inquiry response FHS packet to the device doing the inquiry [1]. This packet contains information regarding the device's address, class etc. which are required for connection establishment.

An inquiry response message is optional; any device that wants to be discovered can send

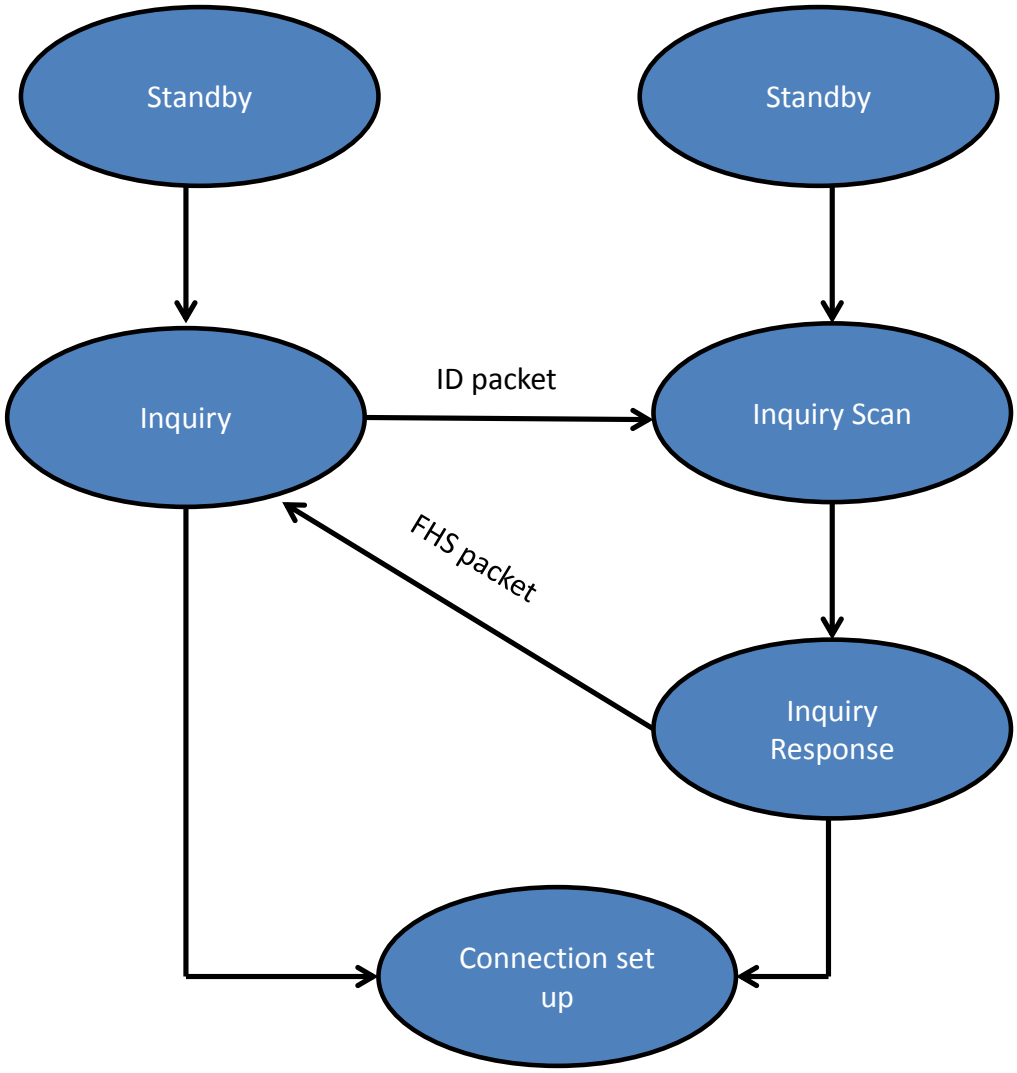

Figure 5.2. Substates in inquiry-based RSSI

the response FHS packet 625 micro seconds after receiving an inquiry message packet. In order to avoid collisions between other responding devices in the range of the master device, Bluetooth uses a random back-off mechanism in the inquiry response mode. According to this mechanism, once a device responds to an inquiry message by sending an FHS packet, it backs off for a random period of time before sending the next response [1]. The device generates an random number between 0 to MAX_RAND and waits for that random number of time slots, where each time slot is $625\mu s$. MAX_RAND is usually 1023 slots for scanning intervals greater than 1.28 seconds. Thus, a device can back-off for a maximum of 1023×625 μs = 0.64s before sending the next response. Therefore, this mechanism gives much less control over the rate at which RSSI values can be obtained.

5.1.1.4 Disadvantages of inquiry mode RSSI for secret key extraction

The inquiry mode RSSI is the only method to obtain *raw* RSSI values for Bluetooth. However, there are several issues associated with this method that are not suitable for RSSI-based secret key extraction. These issues are listed below:

- First, due to the random back-off algorithm in the inquiry response state, and also due to the large time intervals between consecutive inquiry messages, the rate at which the channel is sampled is very low for key extraction.
- Second, Bluetooth does not provide the user any control over the inquiry intervals and hence, these intervals cannot be adjusted by the user to increase the sampling rate.
- Third, the inquiry message is a broadcast message and any device in the proximity of the master can respond to this message. There is no way for the master to select the devices that it needs response from.
- Fourth, the device which wants to be discovered has to be in the discoverable mode in order to respond to an inquiry message. Due to security concerns, current Bluetooth configuration on smart phones allows the device to be in discoverable mode for a maximum of 180 seconds. Considering that the sampling rate with this method is very low, a 180 second time period is too small to obtain RSSI measurements to extract keys of reasonable size.

Therefore in summary, though the inquiry mode gives raw RSSI values, in the current-state-of-the-art, it is impractical to extract keys using this mode.

5.1.2 Connection-based RSSI

Apart from the inquiry mode RSSI, Bluetooth supports another method of obtaining RSSI values in the connection mode. In this mode, the user can obtain RSSI measurement of every data packet received by the device. However, this method requires that a connection be established between two devices and also, it does not return the raw RSSI values. Instead, the RSSI value returned is an indicator of the difference between the raw RSSI value and an ideal power range. In the following section we discuss this in more detail.

5.1.2.1 Golden receive power range

In the connection state, the host controller interface of the BlueZ [3] Bluetooth protocol stack returns an integer value for RSSI. However, unlike in WiFi or the inquiry state, this value is not the exact measured RSSI value, but indicates if the received power level (Rx) is above, below, or within the Golden Receive Power Range (GRPR). This GRPR [1], which is considered as the ideal received power range, is defined using a lower and higher threshold. The lower threshold level corresponds to a received power between -56 dBm and 6 dB above the actual sensitivity of the receiver. The upper threshold level is 20 dB above the lower threshold level with an accuracy of ± 6 dB [2]. As per the Bluetooth specification, a positive or negative RSSI (in dB) indicates an Rx power level above or below the GRPR respectively, while a zero implies that it is ideal (i.e., within GRPR). Figure 5.3 shows the relation between GRPR and RSSI as defined in the Bluetooth specification. The accuracy of this value is hardware specific and the specification just requires the device to be able to tell if the received power lies within the GRPR or not, which affects the reported RSSI value since it is now a relative parameter.

Note that, this method of collecting RSSI values comes with a loss of entropy. This is because, though there might be a change in the *actual* RSSI value, we cannot observe this change unless they are outside the ideal range. Therefore, we will still measure an excessive number of zeroes if the actual RSSI values lie within the GRPR. We observe from our experiments that, for small distances the number of zeroes increases considerably, since the received power lies within the GRPR for a large fraction of time. From our experimental results, we also observe that for smaller distances such as 2 feet in a hallway setting, the number of zeroes could be as high as 90% of the total number of samples.

To summarize, though the connection-based RSSI method gives much control over the type of packets and the sampling rate, it only reports *relative* RSSI measurements as opposed to raw RSSI values from the inquiry-based method. However, we choose to use this method for sampling the Bluetooth channel for secret key extraction, since in the inquiry-based method, the sampling rate is too low to extract secret keys at a reasonable rate.

5.2 Interpolation

Ideally, both Alice and Bob should measure the channel at the exact same time. However, due to the half-duplex mode of operation of the transceivers they are forced to measure the channel one direction at a time. In order to deal with the asymmetry that is caused due to the half-duplex nature of the transceivers, Patwari et al. [15] have used a prequantization

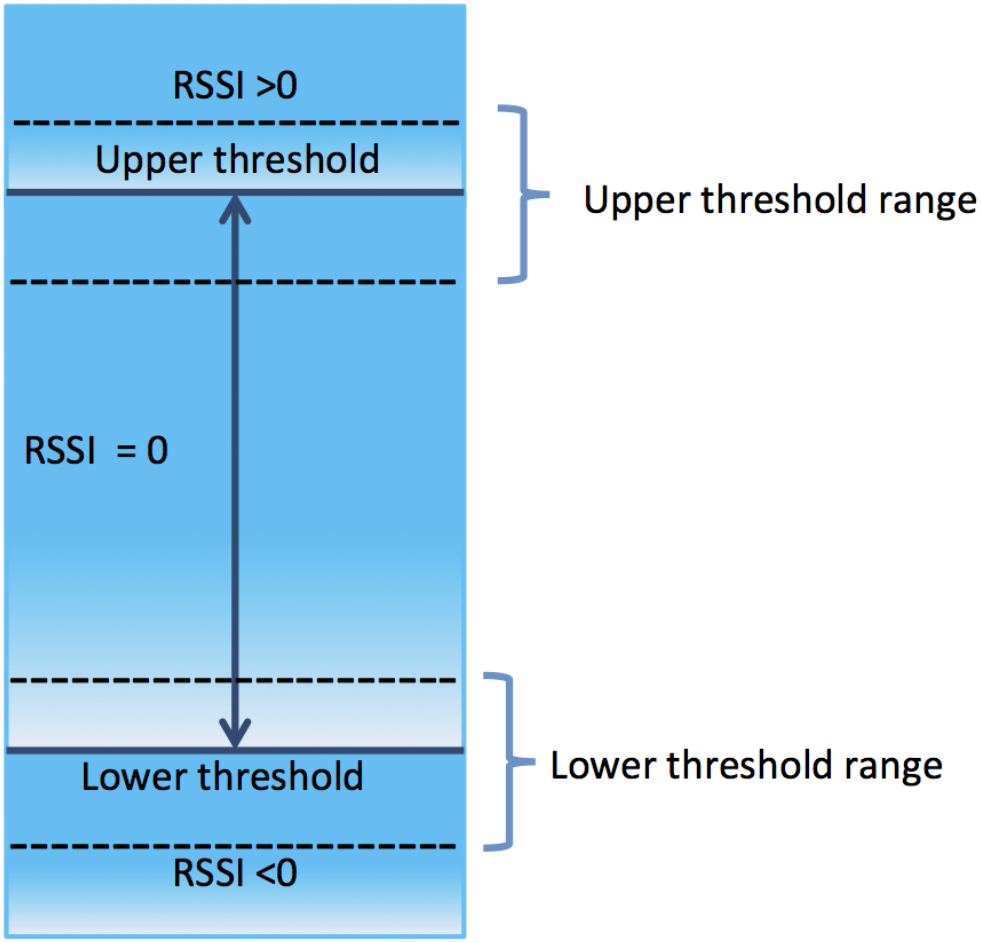

Figure 5.3. Relation between GRPR and RSSI

interpolation step. When Alice and Bob use a time-duplex channel, interpolation addresses the asymmetry that arises due to the time gap in measurement of the time-duplex channel by estimating the measurements of Alice and Bob at common time instant.

If T_R denotes the time delay between two subsequent measurements of Alice or Bob, and $\tau_a(i)$ and $\tau_b(i)$ denote the time instants at which Alice and Bob record their i^{th} measurements, respectively, the fractional sampling offset, μ is calculated as:

$$\mu = \frac{1}{2}\left[\frac{\tau_b(i) - \tau_a(i)}{T_R}\right]; \text{where } \tau_a(i) < \tau_b(i) \tag{5.1}$$

Therefore, if Alice delayed the measuring of her i^{th} value by $(1+\mu)T_R$, and Bob delayed his by $(1-\mu)T_R$, then we would have simultaneous estimates for the ith measurement of Alice and Bob. In our work, we use this interpolation technique as a prequantization stage to reduce the probability of bit mismatches.

CHAPTER 6

IMPLEMENTATION

6.1 Overview

Our key extraction scheme is implemented on two Google Nexus One smart phones. These phones are equipped with the BCM4329EKUBG Broadcom chip [4], which supports Bluetooth Core Specification Version 2.1 with enhanced data rate technology. Both the phones run the Android 2.1 operating system.

Our basic design to establish shared secret keys is as follows: First, Alice and Bob exchange probe packets with each other and measure the RSSI value associated with each received packet. These RSSI values are then quantized [18] individually by using the thresholds to obtain an initial key stream. This bit stream is then subjected to information reconciliation [7] and privacy amplification [10] steps to overcome the bit mismatch and also to increase the entropy, respectively. To collect RSSI measurements for Bluetooth we have designed a protocol using L2CAP [1] sockets. We use the host controller interface of the BlueZ Bluetooth protocol stack to obtain the measured RSSI value for every packet received. We also use a prequantization interpolation step [15] to reduce bit mismatches. The detailed description of the implementation is discussed in the following sections.

6.2 Implementing secret key extraction on smartphones

Google Nexus One phones run on the Android operating system, which is a Linux-based open-source operating system. The Android Software Development Kit (SDK) provides APIs in Java to build applications for the Android platform. The Android SDK also has Bluetooth APIs for some basic higher-level operations which include scanning for other Bluetooth devices in proximity, establishing a connection with one or more devices, transferring data to and from other devices etc. However, the Android SDK does not provide APIs for Bluetooth hardware dependent low-level operations that are required during the

process of secret key extraction. Furthermore, it is best to implement performance-critical operations using C or C++.

In our implementation of secret key extraction using Bluetooth on smartphones, we choose to use a cross compiler to compile our code for the ARM architecture. We use the Bluetooth BlueZ protocol stack for Android. Android-GCC (agcc) is used to compile C code for the ARM architecture and the Android Debug Bridge (ADB) is used for other operations such as copying files to and from the phones, to install applications etc. The ADB enables a host device such as a laptop to communicate with the Android device through USB.

An alternate approach is to use the Android Native Development toolkit (NDK). Although all Android applications run in the Dalvik virtual machine, the NDK allows one to implement performance-critical parts of applications using native-code languages such as C and C++. However, programming the BlueZ protocol stack is significantly simpler than integrating native code using Android NDK and it also provides comparable performance. Thus, we choose to implement secret key extraction for Bluetooth using BlueZ.

6.3 Obtaining inquiry-based RSSI values

As mentioned in Section 5.1.1 , we can use the inquiry mode to obtain raw RSSI values. The host controller interface of the BlueZ protocol stack [3] provides functions to put a device in the inquiry mode with RSSI, which measures the RSSI of any inquiry response message received from other devices. In order to read the raw RSSI values, we use a utility called *Hcidump*, which monitors the Bluetooth HCI data. Using hcidump commands, we can read the raw RSSI values when an inquiry response message is received. To obtain RSSI values for every response packet we first set the inquiry mode to *Inquiry With RSSI*. In order to read raw RSSI values we run the hcidump tool and then use the appropriate HCI functions to start periodic inquiry.

6.4 Obtaining connection-based RSSI values

As already mentioned in our thesis, we use connection-based RSSI values for secret key extraction. In order to sample the channel to obtain these RSSI values, we develop a protocol using L2CAP sockets. According to this protocol, Alice sends a request L2CAP packet to Bob, who measures the RSSI value for this received packet and sends back a reply L2CAP packet for which Alice measures the RSSI value. We use the HCI read RSSI function to read the RSSI value for each received packet. Figure 6.1 shows our

simple protocol and its operation. Alice and Bob continue to exchange the L2CAP packets to obtain time series of RSSI values. We choose to use the L2CAP packet exchange mechanism to collect RSSI samples since L2CAP provides a reliable connection oriented link. L2CAP provides flow control, and handles packet losses through retransmission. In order to support retransmission and flow control, L2CAP uses Supervisory frame (S-frame) apart from Information frames (I-frame) [1]. The information frames are used to carry information payloads, and supervisory frames are sent to acknowledge information frames and request retransmission of information frame, if necessary. Thus, L2CAP provides a connection similar to TCP on the TCP/IP stack. By using L2CAP data packets, Alice and Bob sample the Bluetooth channel to individually collect RSSI values according to the protocol discussed above.

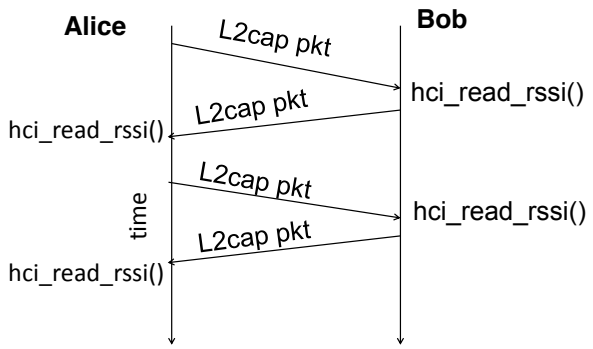

Figure 6.1. L2CAP packet exchange mechanism

CHAPTER 7

EXPERIMENTATION

In this chapter, we discuss the various sets of experiments we have conducted in order to evaluate the performance of secret key extraction using Bluetooth. In this work, we have conducted experiments under two different dynamic environments for varying distances. In this chapter, we first explain our experimental setup, then we present the results of our experiments and evaluate them.

7.1 Experimental setup

In this section, we describe the settings in which we sampled the Bluetooth channels to collect RSSI measurements. Jana et al., in their secret key extraction for WiFi [11] have inferred from their experiments that mobile settings are best suited for secret key extraction. Hence, we choose to conduct our experiments under mobile settings. We conduct several experiments under two different mobile environments for varying distances between the two phones representing Alice and Bob. In each environment, we perform five walk-experiments, where both the phones are carried at normal walking speed and are separated by an average distance of x feet, where $x \in \{2, 5, 10, 20, 30\}$.

Our first environment is a hallway on the third floor of the Merrill Engineering Building on the University of Utah campus. Figure 7.1 shows our experimental set up for the hallway environments and the path taken by Alice and Bob during these experiments.

We conduct a second set of experiments in an outdoor environment across varying terrain, with many trees, strolling people, pets and skate boarders. Alice and Bob are carried at normal walking speed from the Merrill Engineering Building to the Marriott Library on the University of Utah campus. Figure 7.2 shows the trajectory of Alice and Bob in the outdoor environment.

We collect 30000 RSSI samples in each experiment. The sampling rate for hallway settings is around 24Hz whereas, for outdoor settings the sampling rate ranges from 18Hz to 24Hz.

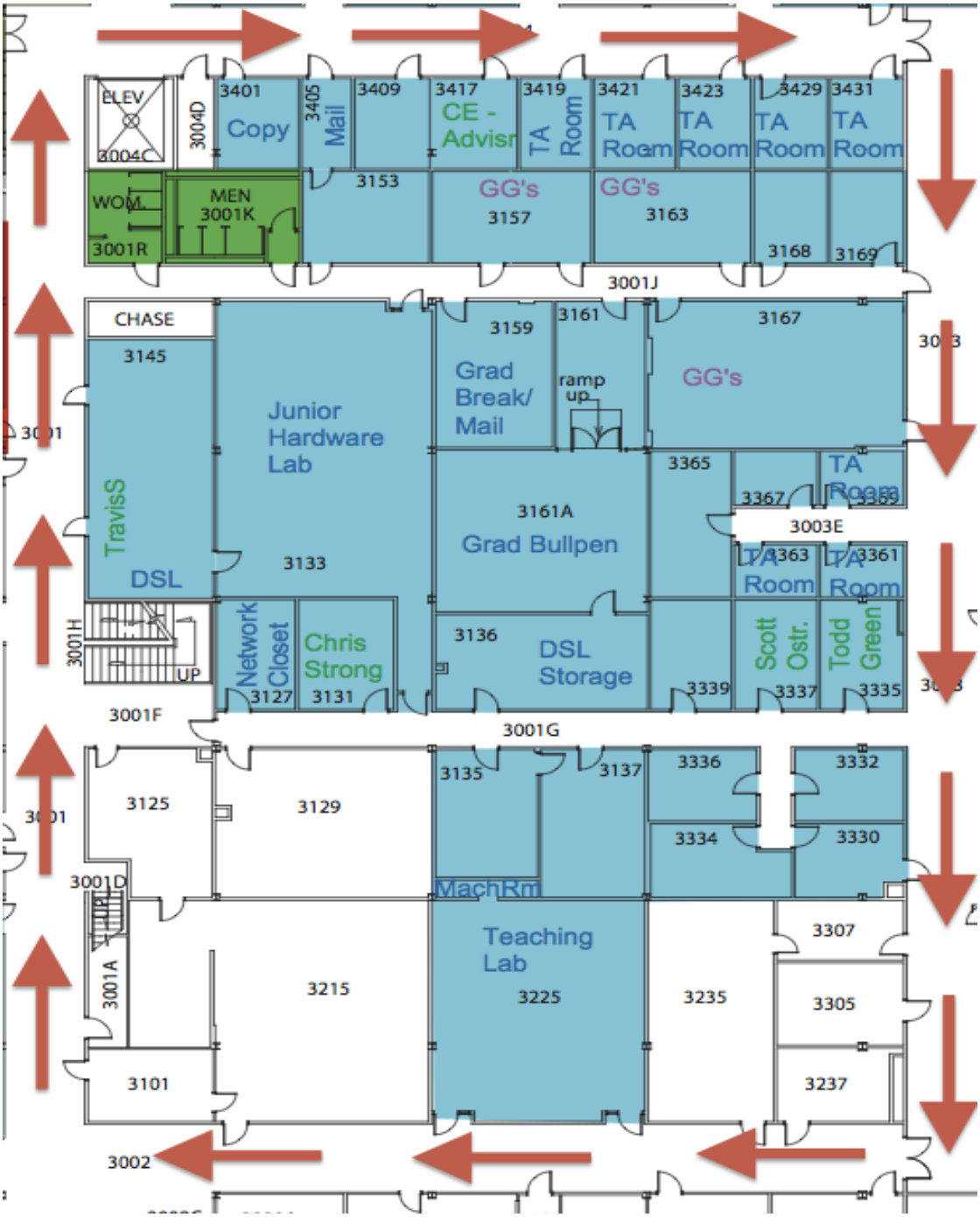

Figure 7.1. Trajectory for hallway experiments

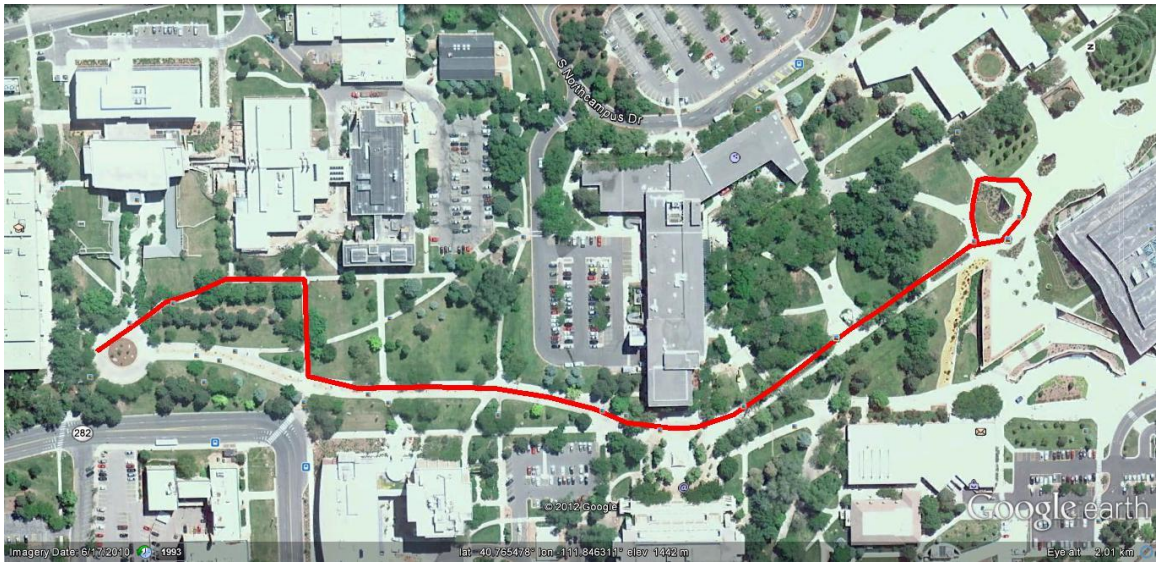

Figure 7.2. Trajectory for outdoor experiments

We find that, under hallway conditions the Signal to Noise Ratio (SNR) is higher compared to outdoor settings. Also, under outdoor settings the packet loss significantly increases with increasing distance and hence, the time taken to collect RSSI samples also increases.

As we have described earlier in Section 2.2, Bluetooth does not provide any control to the *user* over the transmit power. Instead, the link manager automatically adjusts the transmit power of a link based on the received signal strength feedback. We conducted a few experiments and observed that beyond reasonably small distances, Bluetooth switches to maximum transmit power. Furthermore, once it switches to maximum transmit power, it does not revert back to a lower transmit power even when the distance is decreased for the connection. Thus, Google Nexus One phones provide only some rudimentary form of power control.

Table 7.1 summarizes our observations of each experiment.

7.2 Results

In this section we present the results of the experiments conducted under hallway settings and outdoor settings. As mentioned in the previous section, we have done experiments for varying distances and here we present the secret bit rate and bit mismatch rate for each of the experiments, where the bit mismatch rate and secret bit rate are defined as follows:

Table 7.1. Minimum distance between Alice and Bob at which Bluetooth power control chooses the maximum transmit power (3 dBm) in Google Nexus One Smartphones.

Experiment no.	Distance (ft)
1	2.75
2	1.75
3	2.00
4	1.25
5	1.25

- Bit mismatch rate: This is defined as the ratio of number of bits that do not match between Alice and Bob to the total number of bits extracted after the quantization step.
- Secret bit rate: This is defined as the average number of secret bits extracted per probe. It is calculated in terms of the final output bits after accounting for bit losses during information reconciliation and privacy amplification stages.

7.2.1 Key extraction for large distances

7.2.1.1 Bit mismatch rate

Figure 7.3 to Figure 7.6 show the bit mismatch rate as a function of the quantization parameter, α, for varying distances, under hallway and outdoor settings. When we consider the effect of interpolation on bit mismatch rates, we can see that the use of interpolation considerably reduces bit mismatch. As discussed in Section 5.2, interpolation reduces the asymmetry in the measurements by Alice and Bob that arise due to the inability to sample the channel at the same instant of time. We find that interpolation drastically reduces the bit mismatch up to 42% under outdoor settings (for $\alpha = 0.1$ and a distance of 30 feet).

From Figure 7.7, we observe that the bit mismatch rate is much higher under hallway settings when compared to outdoors. This is due to the following reason: Notice from Table 7.2 that the standard deviation of RSSI measurements is much smaller for hallway cases when compared to outdoor cases. As a result, even minor differences in the RSSI measurements of Alice and Bob will cause bit mismatches in the quantized bit stream. Hence we see that hallway settings have higher bit mismatch when compared to outdoors.

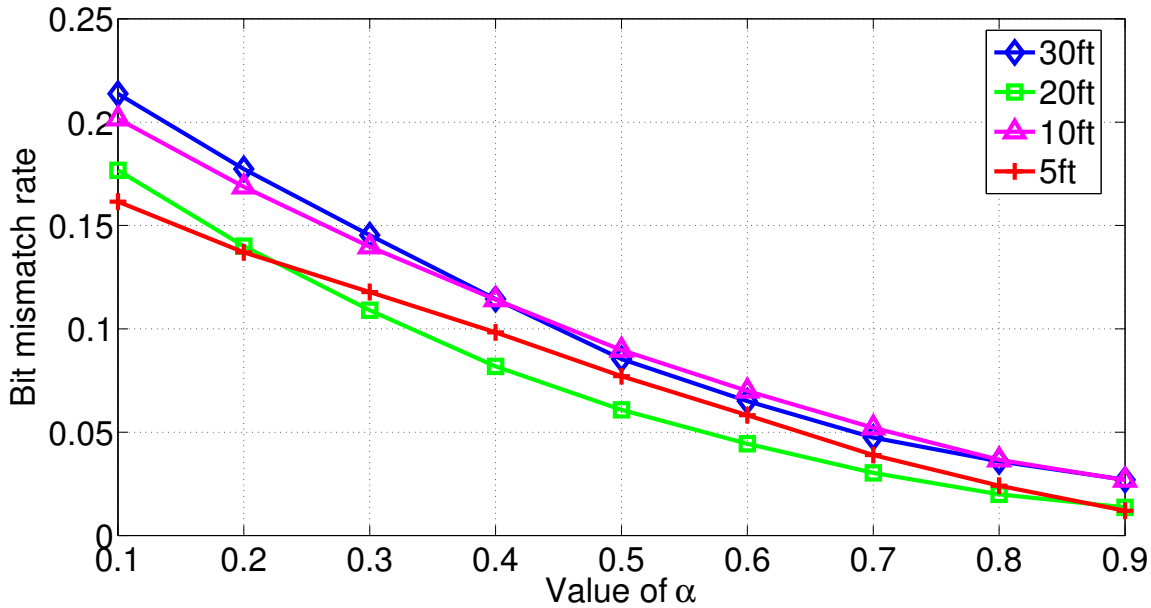

Figure 7.3. Bit mismatch rate as a function of alpha for varying distances (hallway; without interpolation).

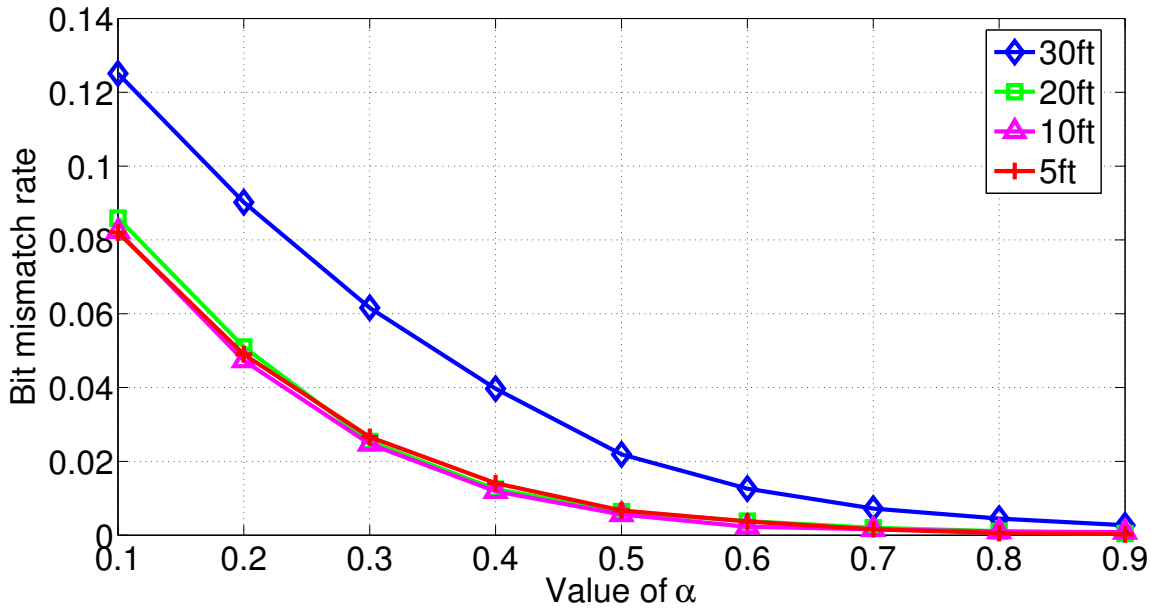

Figure 7.4. Bit mismatch rate as a function of alpha for varying distances (outdoor; without interpolation).

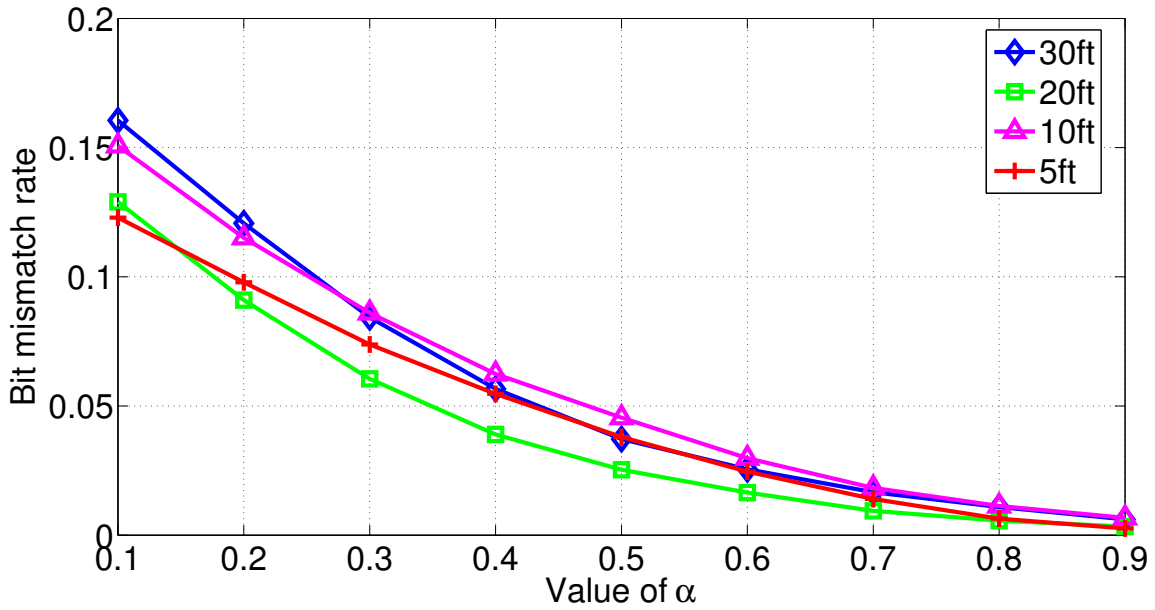

Figure 7.5. Bit mismatch rate as a function of alpha for varying distances (hallway; with interpolation).

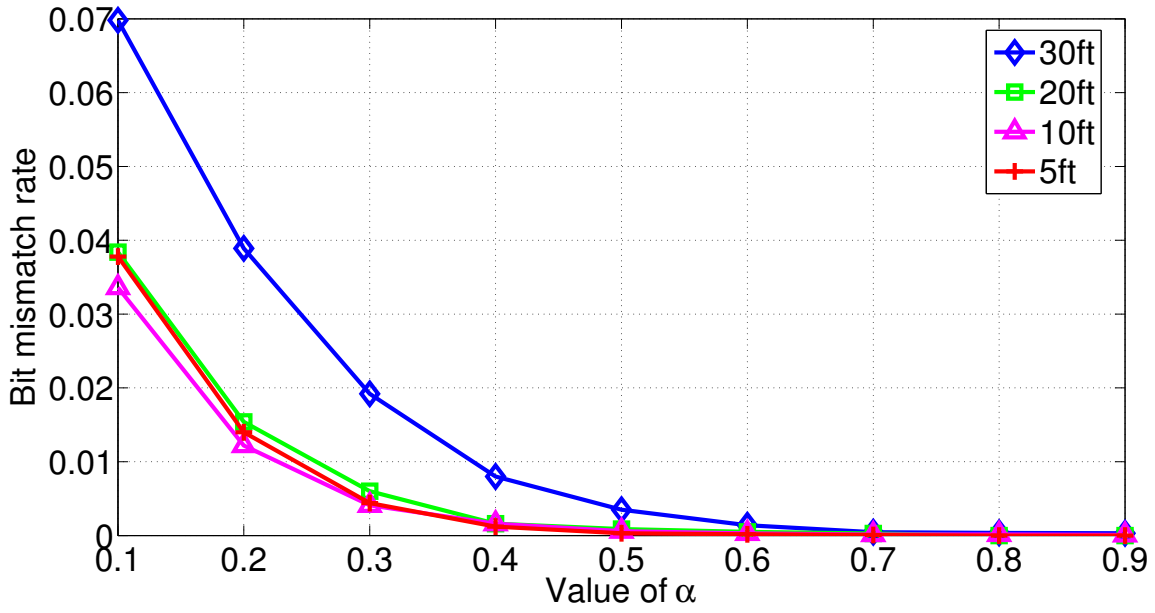

Figure 7.6. Bit mismatch rate as a function of alpha for varying distances (outdoor; with interpolation).

From Figure 7.8 and Figure 7.9, we notice that under outdoor settings, the bit mismatch rate increases with increasing distance. However, for small distances like 2ft, the bit mismatch is high since the number of zeroes increase with decreasing distance. Furthermore, the standard deviation for 2ft is smaller than that for large distances, which also increases the bit mismatch rate. Also, we can see that there is no monotonic increase or decrease in the bit mismatch rate as a function of distance under hallway settings.

Table 7.2. Standard deviation of RSSI measurements averaged over all the quantization blocks.

Distance (ft)	Standard deviation (dB)	Standard deviation (dB)
2	0.2739	4.6984
5	1.9664	5.3959
10	2.4059	5.2477
20	2.8009	4.4812
30	2.4344	2.9787

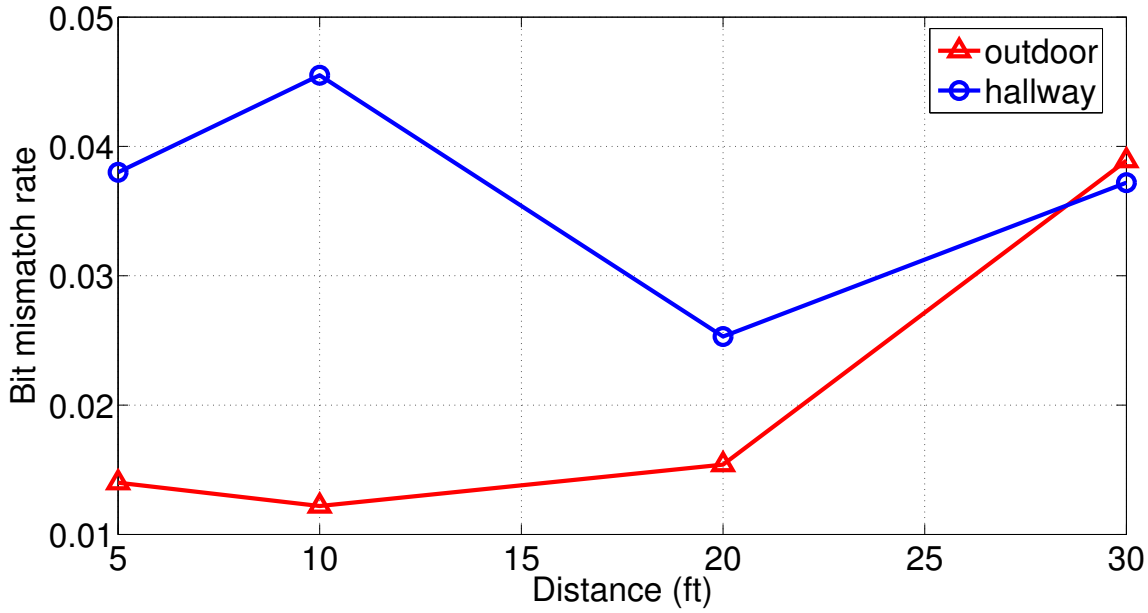

Figure 7.7. Comparison of bit mismatch rate for hallway and outdoor with interpolation.

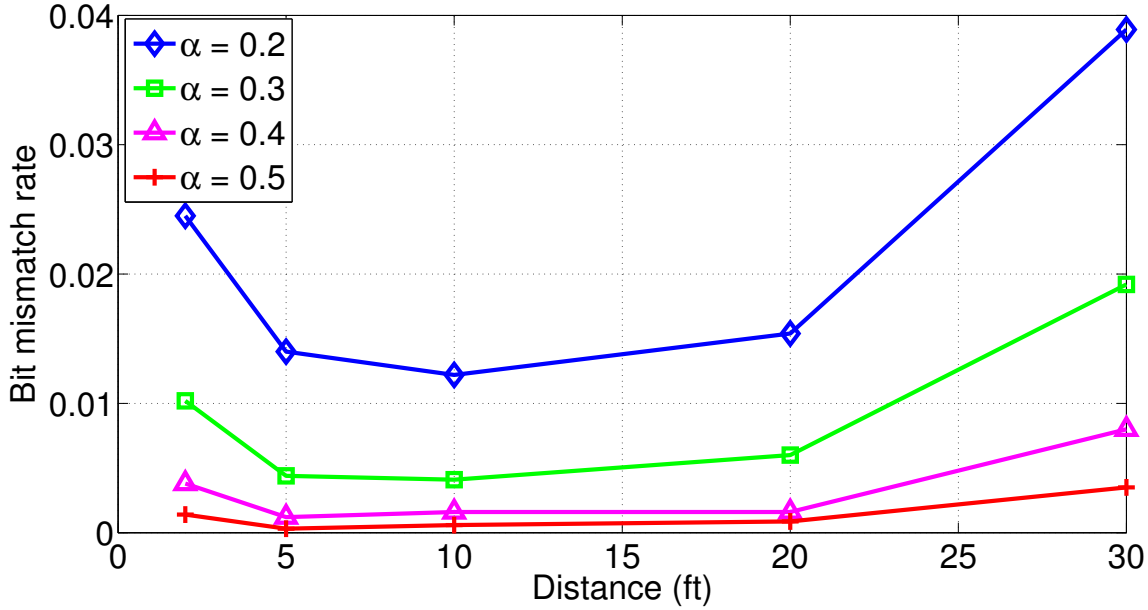

Figure 7.8. Bit mismatch rate as a function of distance for varying alpha values (outdoor; with interpolation).

7.2.1.2 Secret bit rate

Figures 7.10 to 7.13 show the secret bit rate of the Bluetooth interface for varying distances and for varying α values, under hallway and outdoor environments, respectively. We observe that under both outdoor and hallway settings, the secret bit rate is higher with the use of interpolation technique. This is because the interpolation stage reduces the bit mismatch rate, thereby increasing the secret bit rate. If we compare the peak secret bit rates for hallway settings, we can see that the use of interpolation increases the secret bit rate by around 53.8%. Similarly, the use of interpolation increases the secret bit rate by around 25.8% in outdoor conditions.

Next, if we observe the variation in secret bit rate with varying distances from Figure 7.14, we notice that for outdoor settings the secret bit rate decreases with increasing distance. This is due to the fact that signal to noise ratio decreases with increasing distance, which in turn increases the bit mismatch rate with distance. An increase in the bit mismatch rate consequently reduces the secret bit rate. However, under hallway settings we do not observe any clear relation between distance and secret bit rates. This is likely due to the following reasons: The sampling rate for hallways is roughly 24Hz for all distances; in other words, the sampling rate does not change with distance.

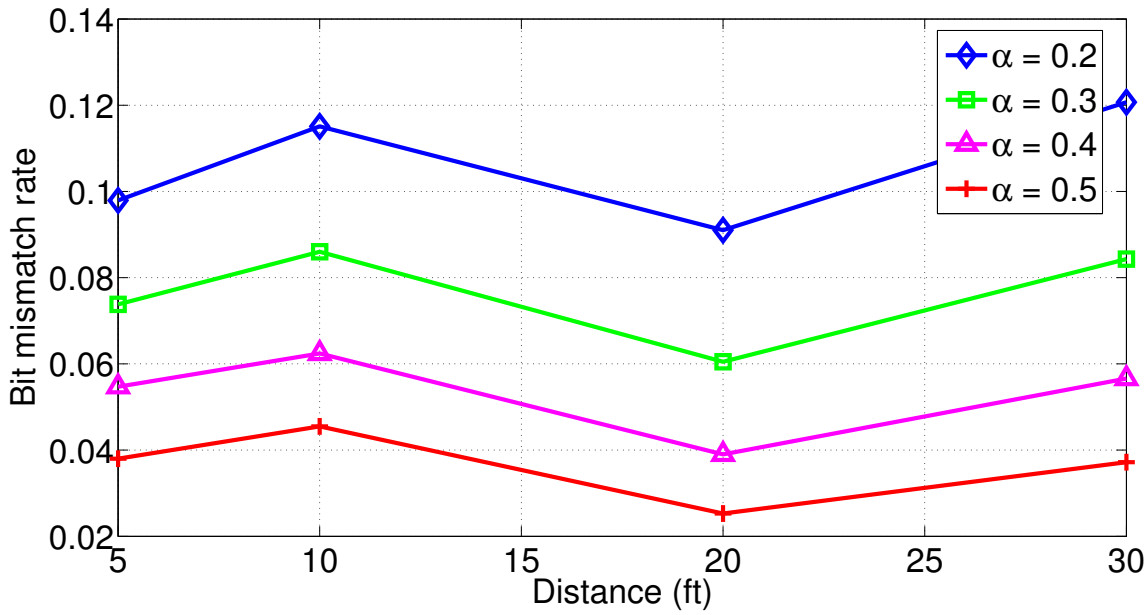

Figure 7.9. Bit mismatch rate as a function of distance for varying alpha values (hallway; with interpolation).

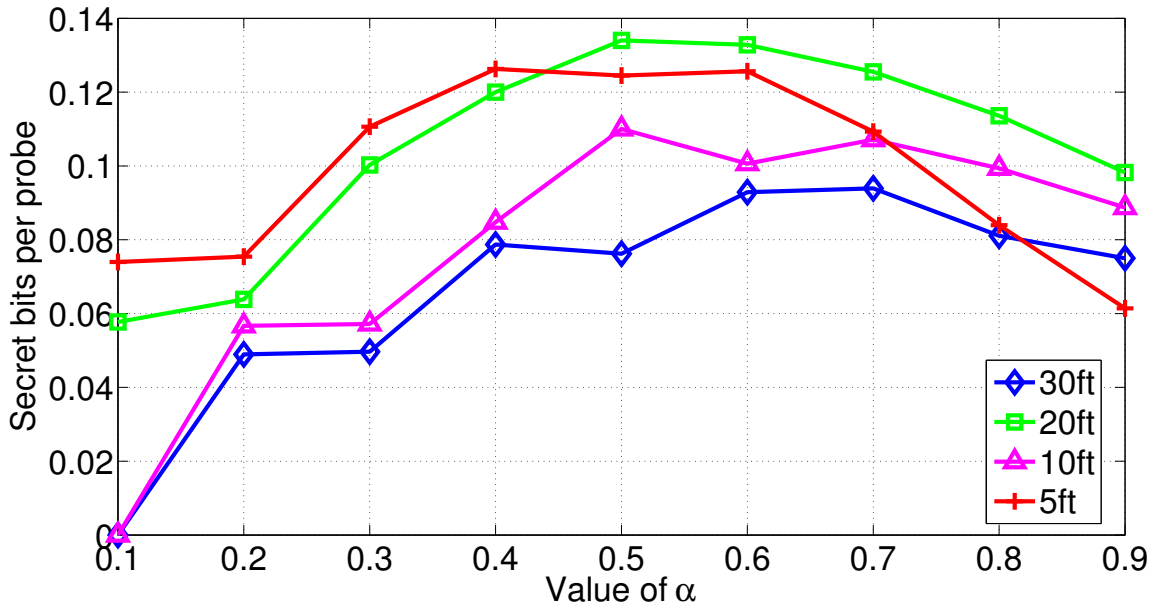

Figure 7.10. Secret bit rate as a function of alpha for varying distances (hallway; without interpolation).

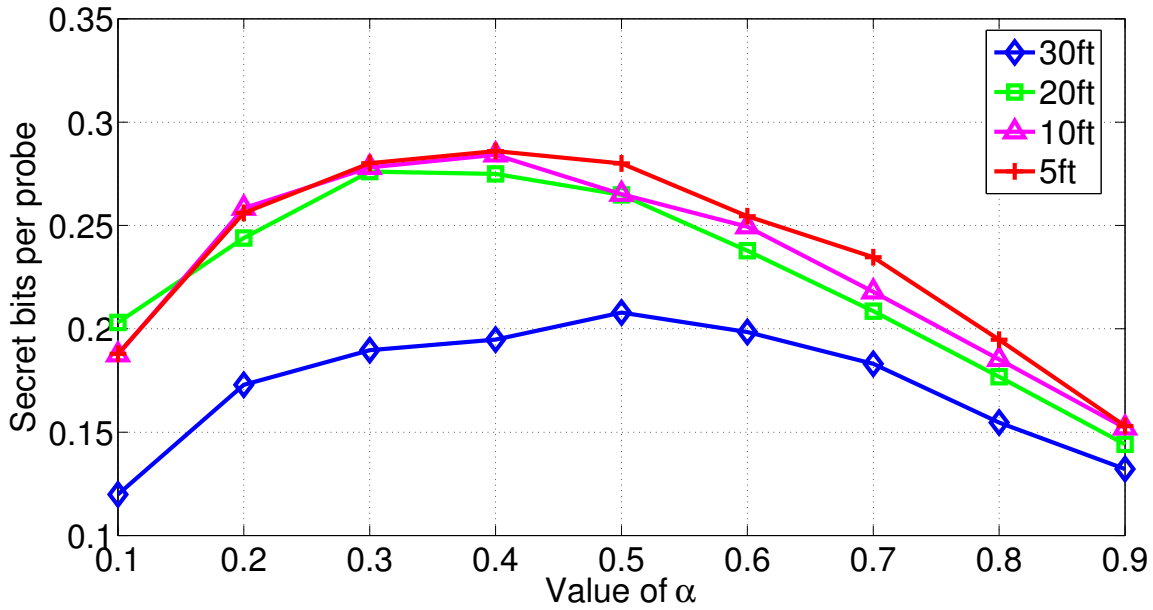

Figure 7.11. Secret bit rate as a function of alpha for varying distances (outdoor; without interpolation).

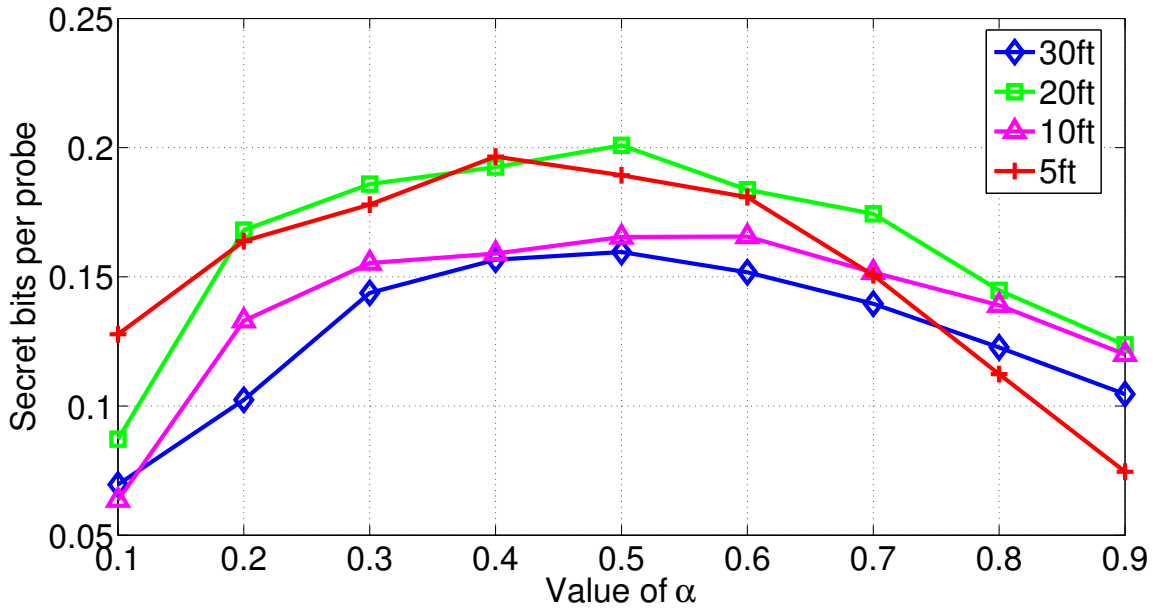

Figure 7.12. Secret bit rate as a function of alpha for varying distances (hallway; with interpolation).

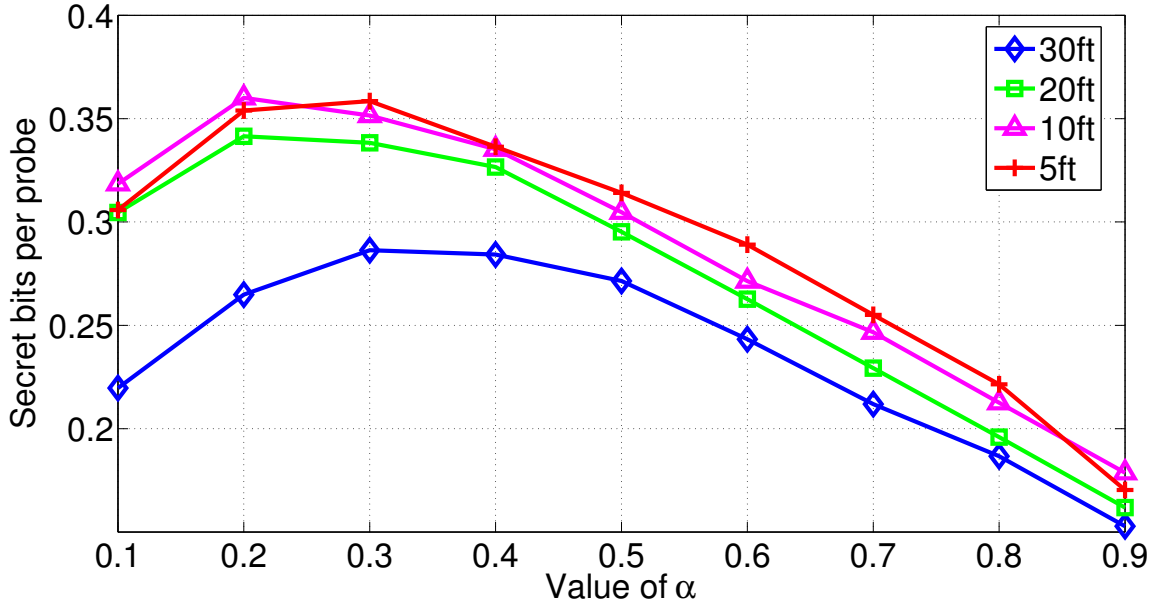

Figure 7.13. Secret bit rate as a function of alpha for varying distances (outdoor; with interpolation).

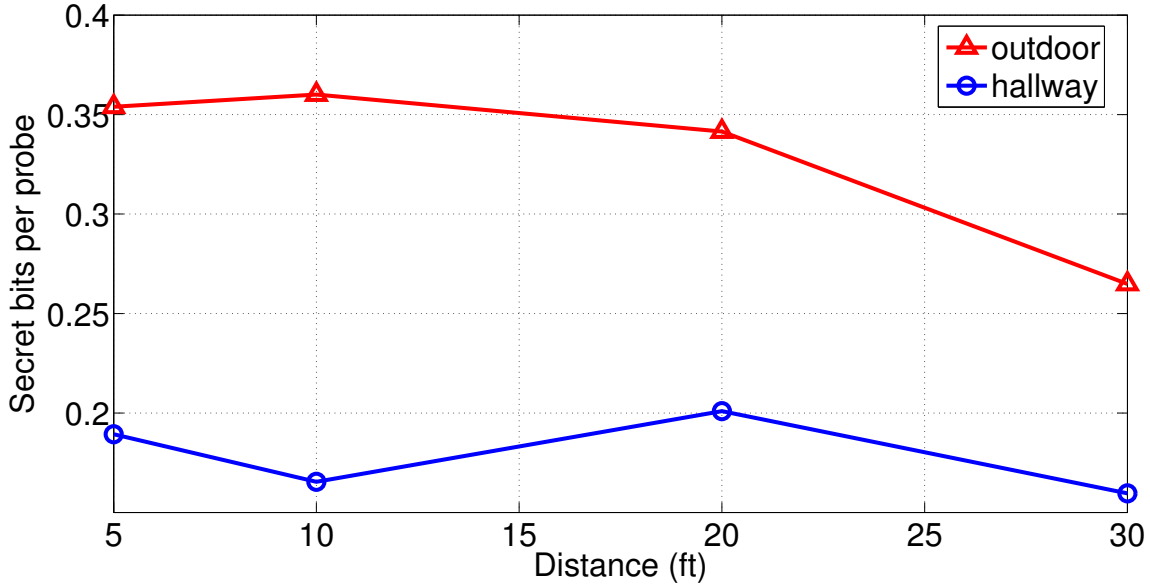

Figure 7.14. Comparison of secret bit rate for hallway and outdoor with interpolation.

This is because the hallway setting can behave like a wave guide where the signal propagation is mostly confined within the hallway due to multiple reflections from the walls, ceiling and the floor [14]. As a result, there is minimal propagation loss with considerable variation in distance. Hence, we do not observe the same trend in hallway settings that we observe under outdoor settings.

If we compare the secret bit rates in hallways with those under outdoor settings, we can observe that outdoor conditions are significantly better for key extraction using Bluetooth. We can see from Figure 7.15 and Figure 7.16 that the maximum secret bit rate with interpolation for hallways is 20%, whereas for outdoors it is 36%.

As mentioned in the previous section, the bit mismatch rate in hallways is higher when compared to the bit mismatch rate outdoors, as a result of which we see higher secret bit rates in outdoor settings as compared to hallways. Also, from Figure 7.17, we can observe that the fraction of zeroes of RSSI values for hallway experiments are considerably higher than outdoor settings, because in hallway settings the received power mostly lies within the GRPR [1]. A large number of zeroes will lead to loss of entropy, thus reducing the secret bit rate.

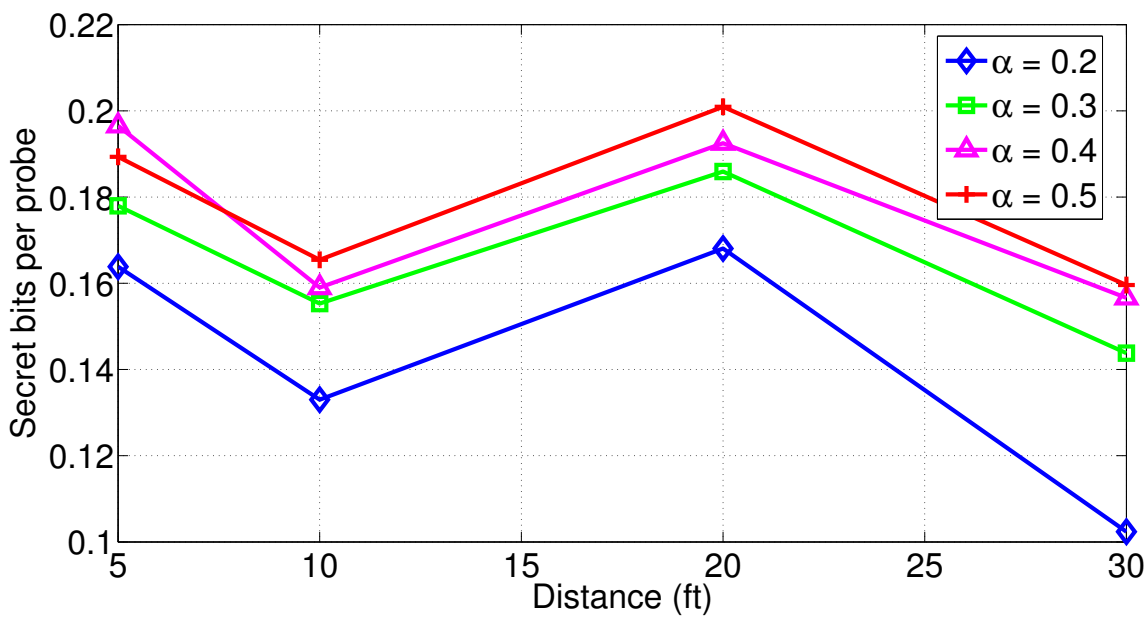

Figure 7.15. Secret bit rate as a function of distance for varying alpha values (hallway; with interpolation).

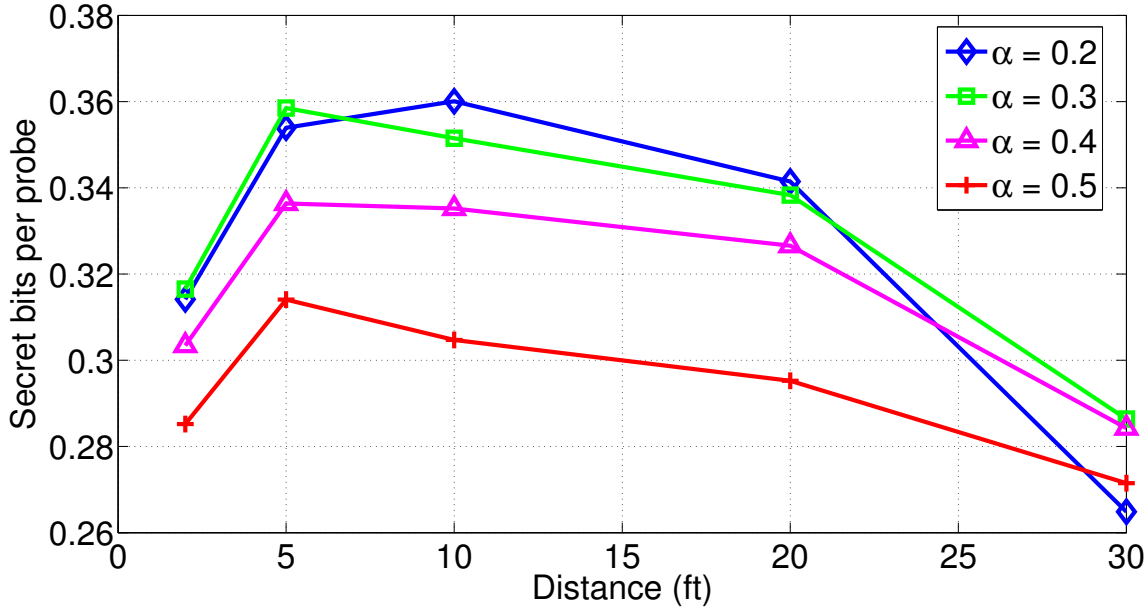

Figure 7.16. Secret bit rate as a function of distance for varying alpha values (outdoor; with interpolation).

7.2.2 Key extraction for small distances

To evaluate secret key extraction for small distances, we conduct experiments maintaining a distance of 2ft between Alice and Bob. We observe that under hallway settings the secret bit rate is as low as 0, whereas under outdoor settings the secret bit rate is around 0.24. The degradation in secret bit rate for hallway settings, in spite of the lower bit mismatch rate can be attributed to the large number of consecutive zeroes. Table 7.3 compares the results of hallway and outdoor settings. Notice that the number of consecutive zeroes is very high under hallway settings, due to which, most of the blocks are discarded during key extracting leading to lower secret bit rates.

7.2.3 Entropy of secret bits

We conduct numerous NIST statistical tests to verify the randomness of the output secret bit streams generated using Bluetooth. Table 7.4 shows the per-bit entropy values for the output secret bit streams that we have extracted in our experiments; notice that all the values are almost close to "1" indicating that there is almost "1" bit of uncertainty associated with each output bit. In Table 7.5 and Table 7.6, we have shown the p-value for eight NIST tests, for different distances. A p-value greater than 0.01 indicates that the input bit sequence is random with a confidence of 99%. Notice that all the p-values are

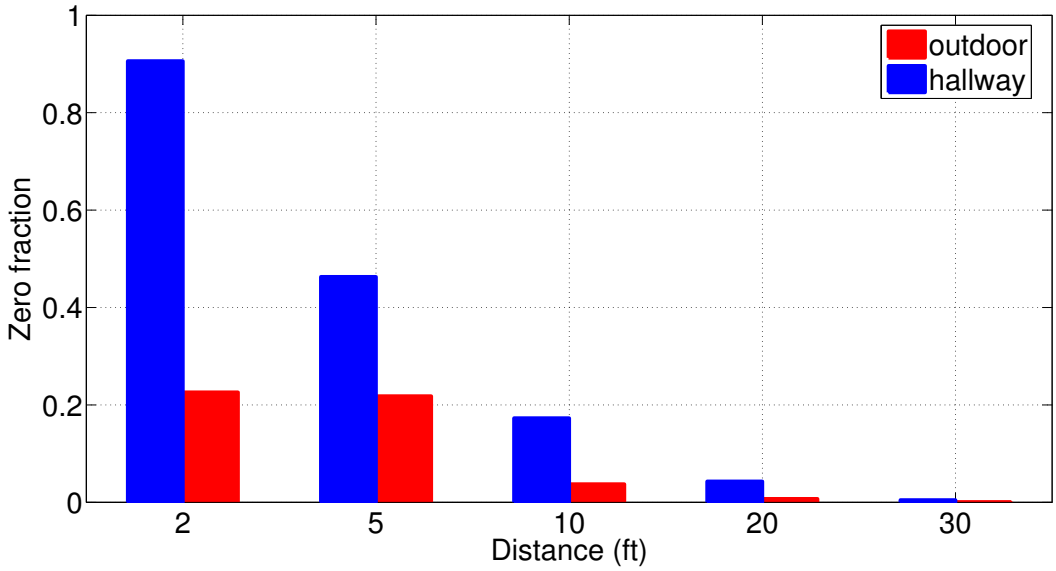

Figure 7.17. Fraction of zeroes versus distance

greater than 0.01, which verifies that all the secret bits streams are random with a very high degree of confidence.

7.2.4 Comparison of WiFi and Bluetooth secret bit rates

Premnath et al. [18] implemented the secret key extraction scheme on Android smartphones for WiFi. We observed that the secret bit rate for WiFi under outdoor settings is comparable to the secret bit rate obtained by Bluetooth under similar settings even though Bluetooth uses lower transmit power. In their outdoor experiments, Premnath et al. used a transmit power of 8 dBm whereas, we use a maximum transmit power of 3 dBm for Bluetooth. Therefore, Bluetooth enables power-efficient secret key extraction.

Table 7.7 shows the secret bit rates of Bluetooth and WiFi for outdoor experiments.

Table 7.3. Secret key extraction at small distances.

Metric	Indoor	Outdoor
Secret bit rate	0.0080	0.2406
Bit mismatch rate	0.0065	0.0261
Expectation of no. of consecutive zero RSSI values	37.6	5.91
Standard deviation of RSSI values of each block	0.514	5.36

Table 7.4. NIST approximate entropy test results

Distance (ft)	Approx. entropy (outdoor)	Approx. entropy (hallway)
30	0.9804	0.9861
20	0.9822	0.9863
10	0.9845	0.9840
5	0.9825	0.9782

Table 7.5. P-values from NIST statistical test suite results for outdoor experiments.

Test	30 ft	20 ft	10 ft	5 ft
Frequency	0.86	0.08	0.51	0.87
Block frequency	0.17	0.18	0.78	0.12
Cumulative sums(Fwd)	0.77	0.14	0.78	0.86
Cumulative sums (Rev)	0.60	0.08	0.63	0.71
Runs	0.80	0.57	0.73	0.30
Longest run of ones	0.69	0.42	0.25	0.48
FFT	1.00	0.14	0.55	0.16
Approx. entropy	0.17	0.06	0.46	0.22
Serial	0.68, 0.64	0.54, 0.60	0.50, 0.14	0.55, 0.40

Table 7.6. P-values from NIST statistical test suite results for indoor experiments.

Test	30 ft	20 ft	10 ft	5 ft
Frequency	0.28	0.55	0.60	0.79
Block frequency	0.20	0.28	0.66	0.33
Cumulative sums(Fwd)	0.54	0.70	0.90	0.79
Cumulative sums (Rev)	0.39	0.29	0.61	0.55
Runs	0.23	0.40	0.55	0.97
Longest run of ones	0.48	0.21	0.92	0.89
FFT	0.96	0.86	0.73	0.97
Approx. entropy	0.48	0.21	0.18	0.16
Serial	0.32, 0.29	0.28, 0.20	0.09, 0.01	0.76, 0.82

Table 7.7. Comparison of secret bit rates - Bluetooth vs WiFi

Setting	**Secret bits per probe**
WiFi 20 ft outdoor	0.2482
Bluetooth 20 ft outdoor	0.2761
Bluetooth 30 ft outdoor	0.2079

CHAPTER 8

RELATED WORK

Premnath et al. [18] have used Android smart phones to extract secret keys using WiFi interface. Also, Croft et al. [8] demonstrated their adaptive ranking-based uncorrelated bit extraction (ARUBE) method in Mobicom 2010, by implementing their method on Android smart phones. However, both these works use WiFi interface for bit extraction. When there is heavy WiFi traffic, there could be excessive medium access delays. As a result, the efficiency of key extraction using WiFi interface could degrade. However, since Bluetooth uses adaptive frequency hopping and selects only those channels which are not occupied or which have comparatively lower traffic, Bluetooth has lesser probability of failing even under heavy traffic conditions.

Existing work [17, 15] use low power devices like telosB nodes for RSSI-based secret key extraction. Premnath et al. [17] have explored the use of multiple frequencies for efficient, high-rate secret key extraction. Their results show that the use of multiple frequencies increases randomness and helps in extracting stronger keys. While their method uses multiple frequencies with the use of *multiple* telosB nodes at Alice and Bob, each Bluetooth device inherently uses multiple frequencies due to the AFH mechanism.

Mathur et al. [12] have designed a scheme to extract secret keys by using RF signals on two USRP nodes. Their scheme does not require the key extracting nodes to have any communication when they sample the channel. Instead, they sample wireless signals coming from a public source such as radio or television signals. In order to record symmetric measurements, their scheme requires the two nodes to be less than half a wave-length away from each other. This implies that if their scheme is used in the 2.4GHz range, the nodes should be separated by less than 6.25 cm, which is impractical. Although the Bluetooth operating range is less when compared to other interfaces like WiFi, we certainly do not have the restriction to place the devices as close. We have also tested our scheme using Bluetooth class 2 devices for up to a distance of 10m to extract secret keys efficiently.

CHAPTER 9

CONCLUSION

In this thesis we have proposed to use the Bluetooth interface for efficient and robust RSSI-based secret key extraction. We have evaluated two methods to collect RSSI samples from the Bluetooth channel, and have determined that the use of inquiry-based method to obtain raw RSSI is not feasible for secret key extraction in the current state-of-the-art. We have also designed a protocol using L2CAP packets to obtain connection-based RSSI values from the Bluetooth channel. We have implemented the key extraction scheme on Google Nexus One smart phones which run the Android 2.1 operating system. In order to evaluate the efficiency of Bluetooth for secret key extraction, we have conducted experiments in two different mobile settings. From the results of our experiments we have observed that outdoor mobile environments are best suited for RSSI-based key extraction using Bluetooth as compared to hallway environments. Furthermore, we have also observed that for small distances such as 2ft, hallway environments can hardly extract any secret key bits. Moreover, although outdoor settings show a decrease in the secret bit rate with increase in distance, there is no clear relationship between distance and secret bit rate in hallway conditions. Lastly, we have observed that although Bluetooth uses lower transmit power, the secret bit rates achieved with Bluetooth in outdoor settings are comparable to the secret bit rates obtained by WiFi in similar settings.

REFERENCES

[1] *Ieee std 802.15.1-2005 ieee standard for information technology telecommunications and information exchange between systems local and metropolitan area networks specific requirements part 15.1: Wireless medium access control (mac) and physical layer (phy) specifications for wireless personal area networks (w pans)*, http://standards.ieee.org/findstds/standard/802.15.1-2005.html.

[2] *Bluetooth specification version 4.0*, https://www.bluetooth.org/Technical/Specifications/adopted.htm.

[3] *Bluez bluetooth stack*, http://www.bluez.org/.

[4] *Broadcom bcm4329 product information*, http://www.broadcom.com/products/Wireless-LAN/802.11-Wireless-LAN-Solutions/BCM4329.

[5] C. Bennett, F. Bessette, G. Brassard, L. Salvail, and J. Smolin, *Experimental quantum cryptography*, Journal of Cryptology, 5 (1992), pp. 3–28.

[6] C. H. Bennett and G. Brassard, *Quantum cryptography: Public key distribution and coin tossing*, in Proceedings of IEEE International Conference on Computers, Systems, and Signal Processing, India, 1984, p. 175.

[7] G. Brassard, L. Salvail, and L. Salvail, *Secret-key reconciliation by public discussion*, 1994, pp. 410–423.

[8] J. Croft, N. Patwari, and S. K. Kasera, *Robust uncorrelated bit extraction methodologies for wireless sensors*, in ACM IPSN, 2010.

[9] N. Golmie, O. Rebala, and N. Chevrollier, *Bluetooth adaptive frequency hopping and scheduling*, in Proceedings of the 2003 IEEE Conference on Military Communications - Volume II, MILCOM'03, Washington, DC, USA, 2003, IEEE Computer Society, pp. 1138–1142.

[10] R. Impagliazzo, L. A. Levin, and M. Luby, *Pseudo-random generation from one-way functions*, in Proceedings of the Twenty-first Annual ACM Symposium on Theory of Computing, STOC '89, New York, NY, USA, 1989, ACM, pp. 12–24.

[11] S. Jana, S. N. Premnath, M. Clark, S. K. Kasera, N. Patwari, S. V. Krishnamurthy, and S. V. Krishnamurthy, *On the effectiveness of secret key extraction from wireless signal strength in real environments.*, in MOBICOM, 2009, pp. 321–332.

[12] S. Mathur, R. Miller, A. Varshavsky, W. Trappe, and N. Mandayam, *Proximate: Proximity-based secure pairing using ambient wireless signals*, in Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services, MobiSys '11, New York, NY, USA, 2011, ACM, pp. 211–224.

[13] S. Mathur, W. Trappe, N. B. Mandayam, C. Ye, A. Reznik, and A. Reznik, *Radio-telepathy: Extracting a secret key from an unauthenticated wireless channel.*, in MOBICOM, 2008, pp. 128–139.

[14] R. Morrow, *Bluetooth: Operation and Use*, McGraw-Hill, Inc., New York, NY, USA, 2002.

[15] N. Patwari, J. Croft, S. Jana, and S. Kasera, *High-rate uncorrelated bit extraction for shared secret key generation from channel measurements*, IEEE Transactions on Mobile Computing, 9 (2010), pp. 17 –30.

[16] L. Pei, R. Chen, J. Liu, T. Tenhunen, H. Kuusniemi, and Y. Chen, *Inquiry-based bluetooth indoor positioning via rssi probability distributions*, in Proceedings of the 2010 Second International Conference on Advances in Satellite and Space Communications, SPACOMM '10, Washington, DC, USA, 2010, IEEE Computer Society, pp. 151–156.

[17] S. Premnath, J. Croft, N. Patwari, and S. Kasera, *Efficient high rate secret key extraction in wireless sensor networks using collaboration*, in ACM Trans. on Sensor Networks (under review), 2012.

[18] S. N. Premnath, S. Jana, J. Croft, P. L. Gowda, M. Clark, S. K. Kasera, N. Patwari, and S. V. Krishnamurthy, *Secret key extraction from wireless signal strength in real environments*, IEEE Transactions on Mobile Computing, 99 (2012).

[19] M. Sprengers and L. Batina, *Speeding up gpu-based password cracking*, in SHARCS, 2012.