November 2016

# INFRASTRUCTURE-FREE SECURE PAIRING OF MOBILE DEVICES

Chunqiu Liu
*University of Massachusetts Amherst*

# INFRASTRUCTURE-FREE SECURE PAIRING OF MOBILE DEVICES

A Thesis Presented

by

CHUNQIU LIU

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL AND COMPUTER ENGINEERING

SEPTEMBER 2016

Electrical and Computer Engineering

# INFRASTRUCTURE-FREE SECURE PAIRING OF MOBILE DEVICES

A Thesis Presented

by

CHUNQIU LIU

Approved as to style and content by:

Tilman Wolf, Chair

Patrick A. Kelly, Member

Christof Paar, Member

C. V. Hollot, Department Head
Electrical and Computer Engineering

# ACKNOWLEDGMENTS

First of all, I would like to express my most sincere appreciation to my advisor Professor Tilman Wolf, who is very knowledgeable and insightful. He always shared expertise and invaluable guidance with me generously on my thesis work. Without his continuous help and encouragement this would not have been possible. He has inspired my interest in the complexity of mechanisms involved in mobile devices communication. His broad professional perspective has introduced me to imagining new ways to enhance the security of mobile devices.

I am also indebted to Professor Patrick A. Kelly for his constructive suggestions that have helped me to tackle signal processing problems successfully that I have faced during my thesis development. In addition, I thank Professor Patrich A. Kelly and Professor Cristof Paar for being my thesis committee members and providing valuable advice on my thesis proposal and defense.

My heartfelt gratitude is also extended to my friends Yiding He, Hongbin Jia, Zheng Shao and Yao He for offering me plenty of help.

Lastly, and most dearly, I would thank my parents for constantly supporting me.

# ABSTRACT

## INFRASTRUCTURE-FREE SECURE PAIRING OF MOBILE DEVICES

SEPTEMBER 2016

CHUNQIU LIU

B.Eng., XI'AN UNIVERSITY OF POSTS AND TELECOMMUNICATIONS

M.S.E.C.E., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor Tilman Wolf

Mobile devices have advanced tremendously during the last ten years and have changed our daily life in various ways. Secure pairing of mobile devices has become a significant issue considering the huge quantity of active mobile device connections and mobile traffic. However, current commonly used file sharing mobile applications rely on servers completely that are always targeted by attackers. In this thesis work, an innovative mechanism is proposed to generate symmetric keys on both mobile devices independently from a shared movement in arbitrary pattern, which means no server needs to be involved and no data exchange needed. A secret wireless-communication channel can then be established with a particular network strategy.

Key words: Mobile devices, secure pairing, features extraction

# TABLE OF CONTENTS

**Page**

**ACKNOWLEDGMENTS** . . . . . **iv**

**ABSTRACT** . . . . . **v**

**LIST OF TABLES** . . . . . **viii**

**LIST OF FIGURES** . . . . . **ix**

**CHAPTER**

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

1.1 Background . . . . . 1
1.2 Motivation . . . . . 2
1.3 Problem Statement . . . . . 4
1.4 Contribution . . . . . 5
1.5 Organisation . . . . . 6

**2. BACKGROUND AND RELATED WORK** . . . . . **7**

2.1 Secure Pairing . . . . . 7
2.2 Literature Review . . . . . 8

**3. RESEARCH DESIGN AND METHODS** . . . . . **13**

3.1 System Architecture . . . . . 13
3.2 Database Module . . . . . 14
3.3 Key Generation Module . . . . . 15

3.3.1 Shazam . . . . . 15
3.3.2 Joint Time-Frequency Analysis . . . . . 16
3.3.3 Short-Time Fourier Transform . . . . . 16
3.3.4 Key Generation Module Design . . . . . 16
3.3.5 Key Generation Module in Practice . . . . . 18

3.4 Networks Communication Module Design . . . . . 23

**4. PROTOTYPE IMPLEMENTATION** . . . . . **25**

4.1 System Functionalities . . . . . 25
4.2 Database Module Implementation . . . . . 26
4.3 Key Generation Module Implementation . . . . . 28

4.3.1 Linear Acceleration Sensor . . . . . 29
4.3.2 Sampling . . . . . 30
4.3.3 Key Sequence . . . . . 32

4.4 Networks Communication Module Implementation . . . . . 33

**5. EVALUATION AND DISCUSSION OF RESULTS** . . . . . **34**

5.1 Security Model . . . . . 34

5.1.1 Security Requirement . . . . . 34
5.1.2 Attacker Capabilities . . . . . 35

5.2 Experiments Design . . . . . 35
5.3 Randomness of Keys . . . . . 36
5.4 Difference between keys by Imitation . . . . . 38
5.5 Performance Discussion . . . . . 38

5.5.1 Sensor Alignment . . . . . 38
5.5.2 Movement Speed . . . . . 39
5.5.3 Movement Pattern . . . . . 39

**6. CONCLUSION** . . . . . **41**

**BIBLIOGRAPHY** . . . . . **42**

# LIST OF TABLES

**Table** **Page**

3.1 STFT Parameters . . . . . 17

4.1 Parameters for Acceleration Sensors in Nexus 7 (2012) . . . . . 29

4.2 Delay Interval Options for Android Sensors . . . . . 31

4.3 FFT Index to Frequency . . . . . 32

5.1 Results of Experiments . . . . . 38

# LIST OF FIGURES

**Figure** **Page**


2.1 ShakeMe Time Domain Full Signal . . . . . 10

2.2 ShakeMe Smoothed Time Domain Signal . . . . . 11

2.3 ShakeMe Extract Features . . . . . 11

3.1 Shake Share Component Modules . . . . . 13

3.2 STFT Window Segments . . . . . 18

3.3 Time Domain Sensor Signals from Two Devices . . . . . 19

3.4 Frequency Domain Sensor Signals from Two Devices . . . . . 20

3.5 512 and 256 Points FFT from Two Devices . . . . . 21

3.6 Peak Frequency Components with Different Windows . . . . . 22

3.7 Networks Communication Module Design . . . . . 23

4.1 Shake Share Functionalities . . . . . 25

4.2 Create Account and Log In Flow Chart . . . . . 27

4.3 Edit Contact Flow Chart . . . . . 28

5.1 Distribution of Key Characters . . . . . 37

5.2 Time Domain Signal When Y and Z Axes Are In the Opposite Direction . . . . . 39

# CHAPTER 1

# INTRODUCTION

## 1.1 Background

PennyStocksLab describes the age we currently live in as "The Golden Mobile Age " [1]. Since the second-generation mobile phone systems emerged in the 1990s [2], the number of mobile devices in the market has increased significantly. This is due to the introduction of smartphones. As users fall in love with their smartphones, mobile devices have become the preferred mode of communication, over the computer-based Internet, as Mary Meeker, a leading venture capital analyst, predicted in her 2008 Annual Internet Trends Report [3]. On average, over 1.8 million smartphones are sold every day - 5 times more than the number of daily newborns in the world [4] - and in October of 2014 over 7.2 billion devices were actively connected, surpassing the population in the world at that time which was between 7.19 and 7.2 billion [5].

Nowadays, it would be impossible to imagine our daily life without mobile devices. They have become an integrated part of our personal life. We have such vast of mobile devices, and we use them to access the Internet and communicate with each other every day. Because of this, the request of connections between mobile devices for people to send messages, post photos and transfer files is increasing substantially. Cisco's "Visual Networking Index (VNI) Global Mobile Data Traffic Forecast Update"has an executive summary about the mobile network in 2015 as following. "More than half a billion (563 million) mobile devices and connections were added in 2015. Smartphones accounted for most of that growth. Global mobile devices and connections in 2015 grew to 7.9 billion, up from 7.3 billion in 2014" [6].

Faced with such massive demands for mobile data, security threats in the process of devices pairing, file sharing or data transferring become increasingly serious. It comes as no surprise that users of mobile devices consider the safety of their privacy as an important factor when they shop for both mobile devices and mobile applications. Application developers and high-tech companies pay great attention and invest significant ongoing money on how to improve security level throughout the procedure of authentication, access control and data confidentiality so that they can protect their customers' private information from being eavesdropped or disclosed by an attacker.

## 1.2 Motivation

Presently, it is common for people to download and make use of mobile applications to send messages, transfer data or share files with each other leveraging P2P networking. There are numerous mobile applications with these functionalities in both iOS Apple Store and Android Google Play markets. Below are several typical widely used mobile applications as examples.

Bump

Bump enables smartphone users to establish connections between devices by bumping them together. When two mobile devices are bumped together, sensors on each device detect the bump and mark down the time and location separately. The Bump application running on each device sends the location of the device, the time stamp of the bump, IP address, and other sensor readings, up to Bump servers [7]. Bump servers listen to the bumps from devices around the world and pairs up devices that bumped at the same small area as well as at the same time. Afterward, Bump servers route information between the two paired devices [8].

Dropbox

The mechanism of Dropbox is simple. It stores all the files on its servers in the cloud and files can be shared with users whoever have the permission, a link, to access the files.

SendAnywhere

When sending a file with SendAnywhere, a 6-digit code will be created automatically by the sending device. A recipient can download that file after manual type in this 6-digit identification number. The best part of SendAnywhere is, unlike Dropbox, it does not save files on its servers for common file transferring [9].

However, these applications I mentioned above have weak points making files that transferred through their servers insecure.

For bump, since it has to send physical information of users to its server, it may be vulnerable to both passive attacks and active attacks. For example, an opponent is able to eavesdrop the location, time stamp and other information as it is being transmitted from a device to one of the servers. After learning the contents of those transmissions, the opponent is able to masquerade as the genuine destination user.

For Dropbox, saving all the files in the cloud would result in its servers being targets for attackers. Attacks such as manipulate or denial of service could be performed by hackers.

For SendAnywhere, the 6-digit number is crucial for file security. However, because it is visible, it is hard to protect the key code during transmitting procedure. Also, since the server controls over the communication, the system is of great potential that susceptible to man-in-the-middle attacks.

There are thousands of file transferring mobile applications on the App Store or Google Play. However, almost all of them have to send customers' information or files up to their on-line servers. Even though it is impossible for me to analyze all of those applications, I cannot say that not a single one can do the work without these strategies. However, I highly doubt that. Even the most secure server that

people trust, iCloud had been attacked in September 2014, and the attack caused users' photos to be released without permission. The consequence was terrible. To overcome this problem, we have to avoid system servers or any trusted authority infrastructure, which with a certain probability of being attacked, during the secure pairing process.

We do need a novel mechanism that initiate s a secure session between two mobile devices easily and securely without sending sensitive information to any server or store data on it. Even more strictly, no any information exchanged at all during devices pairing procedure so that no one has a signal chance to get the key features. In this way, we are able to protect our privacy rights from being violated by the attacks to servers or man-in-the-middle attacks when connecting devices to servers.

## 1.3 Problem Statement

There are various technical challenges in designing and implementing this mobile application. Four of them are the toughest ones that I want to address in this section, and detailed solutions will be proposed in the next few chapters.

The key challenge is how to design a mechanism that enables mobile devices to derive the same encryption key independently and locally which means without any communication between the devices or any server.

Moreover, how to protect the discriminative secret from masquerade or replay attacks, that is the algorithm should be able to determine whether the devices are shaken together or not. For instance, if an opponent monitors the motion pattern that two users shook their devices together, he should not be able to generate the same key as the two users have even though he imitates their gestures.

As for the algorithm, it should able to extract identical features successfully, while being tolerant to the random bits of sensor data.

The last issue is how to deliver an encrypted message to the destination user without sending core contents to a server.

Much more trivial problems that affect the performance of the application such as parameters setting or hardware selection will be discussed in next few chapters as well.

## 1.4 Contribution

It is the purpose of this thesis to ascertain an innovative, practical mobile application system that provides individuals with an easy, fast and secure file sharing service. The emphasis is on securing the pairing mechanism that extracts deterministic features independently of each other on both devices from a shared motion using Joint Time-frequency Analysis, and on network protocol design for communication securely. With this new infrastructure-free system, on the contrary of previous systems, there is no need to send any sensitive information of users to anywhere else for verification or authentication. What is more surprising, there is even no need to exchange any data with another participant during pairing phase. Since no one has preliminary knowledge of the information that constructs the symmetric key that is used to encrypt and decrypt messages, it is able to prevent the risk of revealing users information, especially when sending it to servers or third parties as old mechanisms usually do.

The mobile application that I have designed has features as follows.

Ease of use: users do not have to follow any specific command. As for the user input, just hold two mobile devices in one hand tightly and then shake them together.

Low cost: no additional hardware is required. The only must have element is a linear acceleration sensor, and it is the most basic equipment with low price and is embedded in almost all the mobile devices right now.

Anonymity: servers do not participate in the pairing process and do not store any files submitted from clients as well. Moreover, no data exchange is needed for the key generation process. That means, the involved devices have no access to each other; they generate keys independently.

Security: protected against other devices rather than the destination device to obtain the data or file. The mechanism should be able to distinguish whether the dedicated devices are shaken together or not.

Usefulness: this mechanism should be able to transfer data securely between participating devices. The application encrypts/decrypts content with Advanced Encryption Standard (AES) and transmits information using a secure transmission protocol based on TCP/IP.

## 1.5 Organisation

The rest of this thesis is structured as follows. Chapter 2 presents the background for design scenarios in general, and pertinent literature is also reviewed in this chapter. Chapter 3 introduces methods that are used to establish an infrastructure-free secure pairing system, including system architecture as well as the underlying principle that is used to extract the shared secret from the common movement pattern at both sides and principle for networks communication. Next, Chapter 4 provides the system functionalities and implementation details of the proposed prototype system based on Android platform, such as each parameter setting or functionality called for every module. The evaluation and the performance results of this system are discussed in Chapter 5. Finally, Chapter 6 concludes my thesis work.

# CHAPTER 2

# BACKGROUND AND RELATED WORK

## 2.1 Secure Pairing

Before a wireless-communication channel is established, no matter it is for short-lived interactions or longer ones, device pairing is required to mutual authorize the participant devices for security reason. Typically, communication technologies such as NFC, Bluetooth are commonly used for mobile devices pairing. These methods have disadvantages that may affect the security level, for example, must stay within personal area networks, device information exchange needed or personal identification number(PIN) is visible.

To overcome these drawbacks, concealing the keys are needed. Keys shall be symmetric considering the objective of this thesis is to pairing devices without a public-key infrastructure. To strengthen symmetric keys, true random numbers are better than pseudo random numbers. One way to implement a true random number generator using a mobile device is combining the software with hardware, that is importing hardware physical variables with user-device interactions into software program.

For mobile devices, their various sensors are perfect resources to get true random numbers. However, environmental sensors are not able to distinct devices within the same small area. I choose accelerometer as the sensor to get true random numbers because it is very cheap and almost every mobile device in recent years is equipped with 3-axis accelerometers. This obviously leads me to choose simultaneous shaking as the kinetic user interaction input. Since shaking process gives two involve devices similar

motion experience, that almost the same movement patterns produce approximately sensor readings [10]. Besides, shaking is familiar to users, no need to learn.

## 2.2 Literature Review

The idea of pairing small mobile devices by a shared shaking motion was first proposed in "Smart-Its Friends ". The approach was insecure since it broadcasted movement features without authentication in the interaction [11].

Closely related is [11] carried by Rene Mayrhofer and Hans Gellersen . Two concrete methods for providing secure pairing of mobile devices were demonstrated in this paper that were ShaVe and ShaCk. "ShaVe "was designed for key verification but need to exchange sensor readings; "ShaCk "was introduced as a protocol for generating identical key by matching features that extracted from acceleration characteristics, however, the drawback of this method was the demand of transmitting candidate feature vectors from one device to the other one.

Another relevant work is [12]. However, their conclusion is based on an assumption that two accelerator are start at same time. Actually, they manually synchronize the data for analysis. They deal with signals in time domain, and they claimed in this paper that "frequency domain is not suitable for this situation since the high cross-convariance of the frequency spectrum ".

The prior work that probably has the closest relation to mine is [10]. The scenarios that generate key from a shared motion is same, however, they extract key features in time domain using low-pass filter and quantization. They provide test result with (76%, 4%) however, I highly doubt that negative test class is not designed in an appropriate way. As they proposed in their paper, the positive test is performed by 50 subjects shaking two devices in one hand for 5 times. While, the negative test is designed as choosing two test data from different subjects randomly to see if the keys are same.

For my prospective, the right way for designing the negative test class should be as following.

Test A) One person shake two devices together with one device in each hand and during the shaking process try his best to maintain the same motion of two hands.

Test B) Two person shake two devices with one device in each hand. Person A shakes device A in an arbitrary movement pattern. While person B try his best to imitate the pattern of A.

In this way, we can conclude that the approach is able to distinguish different motion even the two movement pattern are similar but not identical. That is vital to test the approach, since the motion pattern is the only information that used to construct the secret key. We have to make sure that an attacher is not able to get same key even he imitate the motion at the same time or he copy the motion after record it. Anyway, only randomly check two different motion generates different key is persuasiveness. So that even though the result of negative class test is only 4% can not guarantee the security of the secret level.

I really curious about why they still got 4% for their negative test class even with inappropriate designed test method, so I reproduce their experiment. Here is an example for positive test class with data I collected by my devices using their settings.

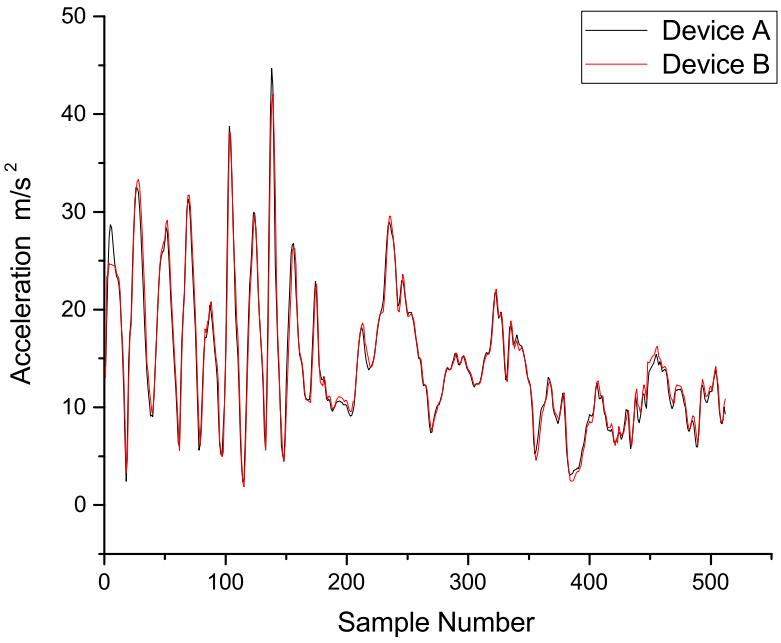


**Figure 2.1.** ShakeMe Time Domain Full Signal

The figure above shows the time domain signal collected on two devices. Even though the acceleration pattern is quite similar, the descriptive statistics variables are not suitable for directly use. So they pass the signal to a box filter with kernel size = 40. I perform the signal smooth with adjacent averaging method in stead using Origin Pro, the points of window is set to 40. The following figure shows the smoothed signals which is qualified to reproduce their approach.

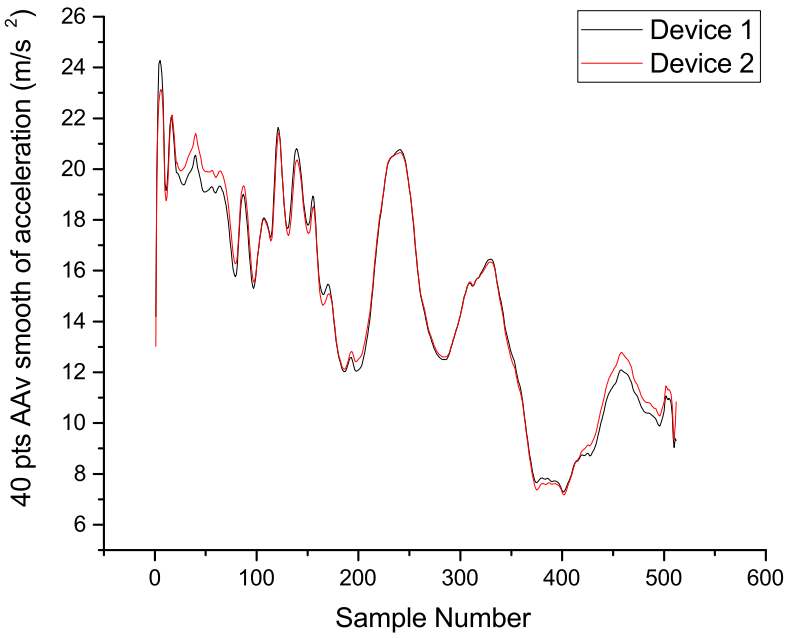


**Figure 2.2.** ShakeMe Smoothed Time Domain Signal

It is clear that with this smooth process, a lot of movement information is lost. That leads to increase the failure rate of negative test class.

| Features | Device A | Device B | Normalized A | Normalized B | Binary A | Binary B |
|---|---|---|---|---|---|---|
| Number of Peaks | 17 | 17 | 0.95724 | 0.96644 | 1 | 1 |
| rms | 15.24767 | 15.36536 | 0.86468 | 0.87916 | 0.111 | 0.111 |
| Mean | 14.65322 | 14.78175 | 0.83328 | 0.848 | 0.111 | 0.111 |
| Variance | 17.80946 | 17.62863 | 1 | 1 | 1 | 1 |
| Skewness | -0.02334 | -0.06289 | 0.05805 | 0.05543 | 0 | 0 |
| Kurtosis | -1.12228 | -1.10113 | 0 | 0 | 0 | 0 |

**Figure 2.3.** ShakeMe Extract Features

The table above shows the results according to their approach. First, those 10 features are not related with time, so it leads to wrong negative class test results. For example, if one signal is time domain translation of anther one, according to their method, the extracted features will be same. It is wrong because, actually they are not from the same motion. Dealing signal only in time domain will lost motion

information. While I use STFT, which is a basic method of Joint Time Frequency Analysis. It is able to get information of signal in both time and frequency domain.

Anther thing is that put a set of normalized numbers through a decimal-to-binary quantilizer makes their binary string is not strong enough to be used as cryptographic key. Since the binary results have high probability to be 0 or 1, that is less randomly.

# CHAPTER 3

# RESEARCH DESIGN AND METHODS

## 3.1 System Architecture

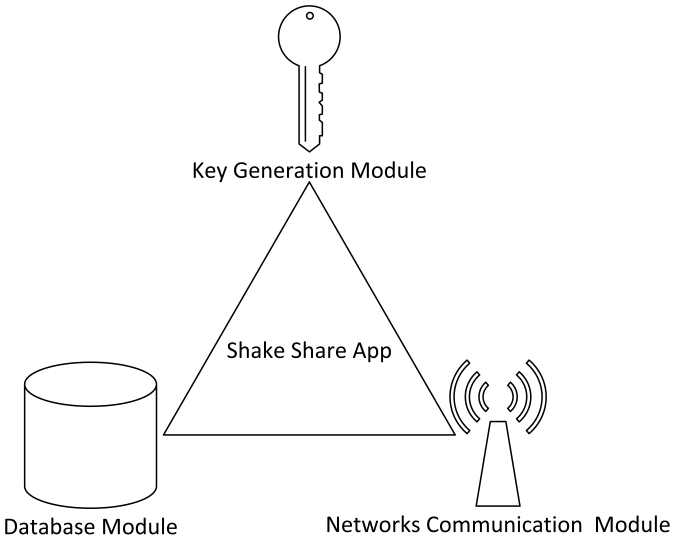


**Figure 3.1.** Shake Share Component Modules

The application system that this thesis proposes mainly consists of three modules: the database module, key generation module and networks communication module.

The database module contains confidential information of both users and their contacts, especially the generated keys and the IP addresses of users. The design of database will be stated in the second section of this chapter.

The second component of the application system, the key generation module, is the most important and innovative part of this thesis. The key objective of this module is to construct a symmetric key with two shaken mobile devices that are

independent of each other based on a common movement pattern. The underlying principle of this independent key generation algorithm will be illustrated in the third section of this chapter.

The third component is as the purpose of securing communication over the network. This application uses a particular strategy based on TCP/IP to send and receive messages through the Internet. Details of the networks protocol will be demonstrated in the fourth section of this chapter.

## 3.2 Database Module

The module is divided into two parts. One is a local SQLite database for storing the username and password combination for user login and the contact list for each user with the secret key bits corresponding to the contact names. These pieces of information are designed to be stored locally on devices because the cryptographic key is crucial for the entire approach to guarantee security. If the secrets are stored in the cloud, attackers may have a chance to get it, thus making the approach vulnerable. This reasoning does not apply to storing user login information locally. Local storage implies that an attacker cannot successfully login from another device even though the attacker has the correct username and password combination. In other words, anther entity cannot access the essential information without the user's device. It is user's responsibility to keep his/her devices safe and not lose them, otherwise the user lose all the data.

The other part of the database module is the server database which stores a table of usernames and IP address as variables. No significant information is stored in the cloud so that even though an attacker discloses the information, one can do nothing with this useless information. An attacker may modify the IP address to his/her own to receive messages for the attacked account, but the attacker cannot decrypt the cryptograph without the encrypted key. This kind of attack that denies the user

service is the only thing that an attacker can do with the information that is stored in the server. However, even though the messages are lost, security has not been damaged, since their content is not released.

## 3.3 Key Generation Module

The main idea of key generation module is extracting identical variables, which turn out to be the index numbers of peak frequency components, from acceleration samples using Short Time Fourier Transform (STFT). First, the feasibility of this approach needs to be generally examined.

### 3.3.1 Shazam

The idea of utilising peak frequency component to construct the desired symmetric key is inspired from Shazam, an application that identifies the song that an user-entered short music clip comes from. I lend this from lecture notes of Prof.Kelly?s course, introduction to electrical and computer engineering. Shazam works by matching the patterns of peak frequencies in spectrograms. When its server receives a clip, it computes the spectrogram of the clip first. For each time interval,it identifies the peak frequency that is the frequency component having maximum magnitude. Then the pattern of peak frequencies vs. time serves as a unique "fingerprint "of this clip and be compared with all the peak patterns of over 1 million songs stored on its servers.

Based on the super performance of Shazam, it is convincing that peak frequencies patterns are able to distinguish different signals. Meanwhile conversely, if two signals are generated from same movement pattern, peak frequency components shall be fall in the same frequency.

### 3.3.2 Joint Time-Frequency Analysis

It is known that Joint Time-Frequency Analysis (JTFA) is a powerful method to analysis non-stationary signal for it describes instantaneous frequency changes over time.

To derive a common secrete by measuring the acceleration while shaking the devices, unlike the prior art, JTFA is implemented in this approach, especially short-time Fourier transform (STFT). The reason is the induced information signal, which is linear acceleration sensor data that collected during shaking processes, essentially is non-stationary signal. That means its frequency is time-varying, so classical signal processing methods based on Fourier transform are not enough to solve this problem.

### 3.3.3 Short-Time Fourier Transform

The most basic and common methods of JTFA is the short-time Fourier transform (STFT). To achieve STFT, non-stationary signals are segmented by short of time domain windowing, so that they turn into a series of short-time stationary signal. Therefore, the short-time Fourier transform is also named as windowed Fourier transform.

### 3.3.4 Key Generation Module Design

Before performing STFT, preprocessing raw data collected by linear acceleration sensors is needed. Mobile devices may have diverse coordinate spaces by different manufacturers and various models. That means the two involved devices may not hold and shaken in a 3-dimension corresponding position. For instance, the x axis of the device A is as same direction as the y axis of the device B. The first task to deal with is sensor spatial alignment. A solution is reduction dimensions with the equation of $\sqrt{x^2 + y^2 + z^2}$. By doing so, three dimensions shaking processes are reduced to be one-dimension.

After preprocessing the sensor data, the one-dimensional data is divided into segments of equal length and then Fast Fourier Transform (FFT) is performed on each segment to get magnitudes for the components at different frequencies. Based on the results of FFT, position of peak frequency component, that is the index number of FFT results, is extracted to construct the desired symmetric key.

Other information is not working other than the index number of FFT results. For instance, the peak magnitude, descriptive statistic variables of the FFT results like mean, variance or Kurtosis.

It is possible that using STFT to compute the key. Next task is setting STFT parameters, including window type, window length, window correction factor, FFT length and overlap. However, one of the weak points of STFT is the resolution conflict of time and frequency domain. The wider time window, the higher frequency resolution is, however, lower time resolution is. After analyze data samples using Origin 2016, which is a scientific data analysis software, set parameters as the following table is the optimum option.

**Table 3.1.** STFT Parameters

| FFT Length | Window Length | Overlap | Window Type | Window Correction |
|---|---|---|---|---|
| 128 | = FFT Length | 0 | Rectangular | None |

FFT length shorter than 128 fails to work. Overlap is set to zero to avoid correlation between each segment. Basically, these parameters are set to get the optimum results with the least complexity for java programming.

Only few numbers is definitely not enough for a secure key strength. To get more bits, divide the full signal by different window lengths. Right now, the full signal is first converted to frequency domain using FFT directly to find the main frequency component, and then divided by 3 different length of windows, 512, 256 and 128. To avoid correlation as much as possible between each window length, moving window

slightly in time domain so that for each window FFT is computed with samples in different time period. Full signal is divided as the below figure.

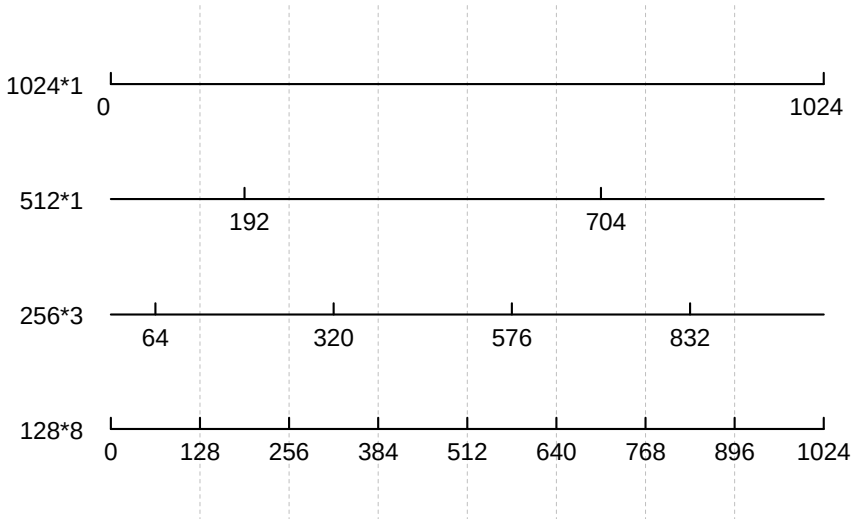


**Figure 3.2.** STFT Window Segments

### 3.3.5 Key Generation Module in Practice

This principle works well in practice. For instance, after shaking two devices together for 5.12 second with sampling rate at 200 samples per second, the 1024 samples collected by linear acceleration sensor in time domain are shown as Figure 3.3.

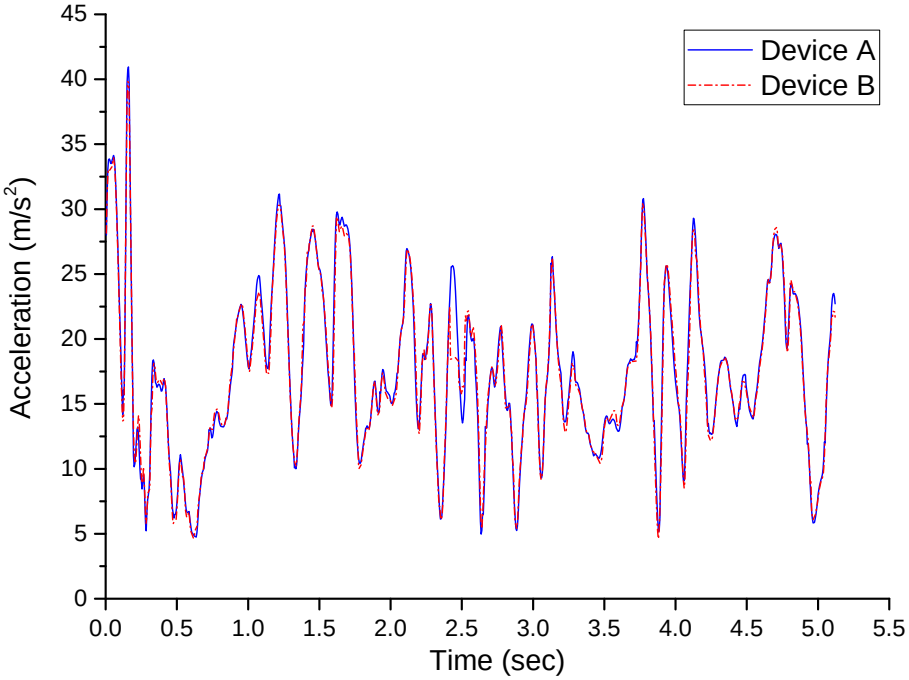


**Figure 3.3.** Time Domain Sensor Signals from Two Devices

In Figure 3.3, clearly from time domain, the sensor signals are almost but not identical same. Pass these two entire signal sets to FFT for converting time domain into frequency domain. For each window segment, leave out the DC component first, which is the zeroth value, then find the peak value in first half of data considering FFT results is symmetric. The result is shown as Figure 3.4.

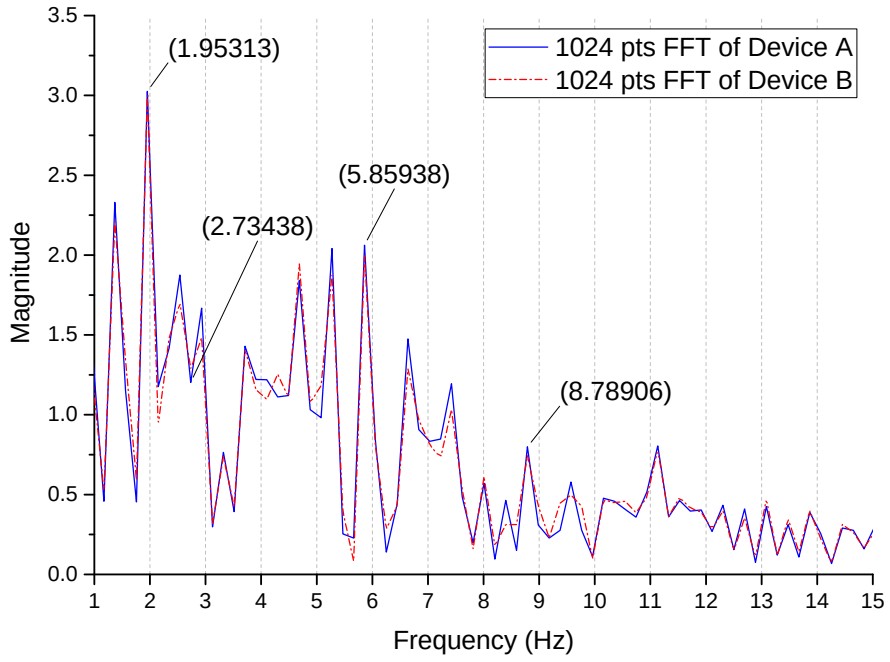


**Figure 3.4.** Frequency Domain Sensor Signals from Two Devices

In 3.4, the frequency components span the range from 0 to 100. However, when frequency is larger than 9 Hz, FFT results are very small. Tests shown that the frequency of a normal human shaking a mobile device is under 9 Hz using Nexus 2012. And leaving out low frequency components is necessary, since if the movement is too slow, it is very easy for attacker to copy the motion pattern. The frequency range that considered in this approach is (2.73438 Hz, 8.78906 Hz). Therefore, even 1.95313 Hz is the peak frequency within the entire range, it will not be considered. 5.85938 Hz will be record as the peak frequency component.

For 512, 256 and 128 points segments, the observation is focus on (2.34375 Hz, 8.98438 Hz) to cover the considered range. For 256/128-STFT, the discrete signal samples are split into groups of 256/128 points, that is, first group of 256/128 signal samples is pass through an FFT then FFT is performed over and over again to the

rest of every 256/128 sensor data to get the magnitudes of frequency components for each time interval. The results of 512-FFT and 128-STFT are shown as Figure 3.5.

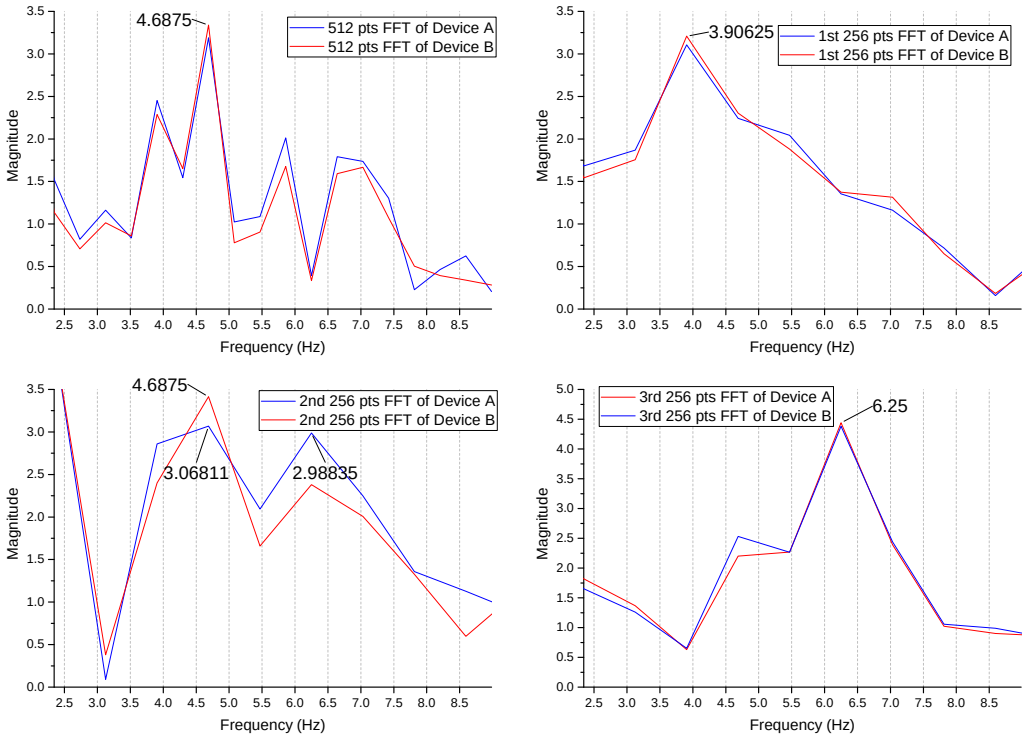


**Figure 3.5.** 512 and 256 Points FFT from Two Devices

Similarly, 128-STFT results are (3.125,3.125,4.6875, 6.25,4.6875,4.6875,4.6875,4.6875). Compare all the peak frequency components in different segments shown as the figure below.

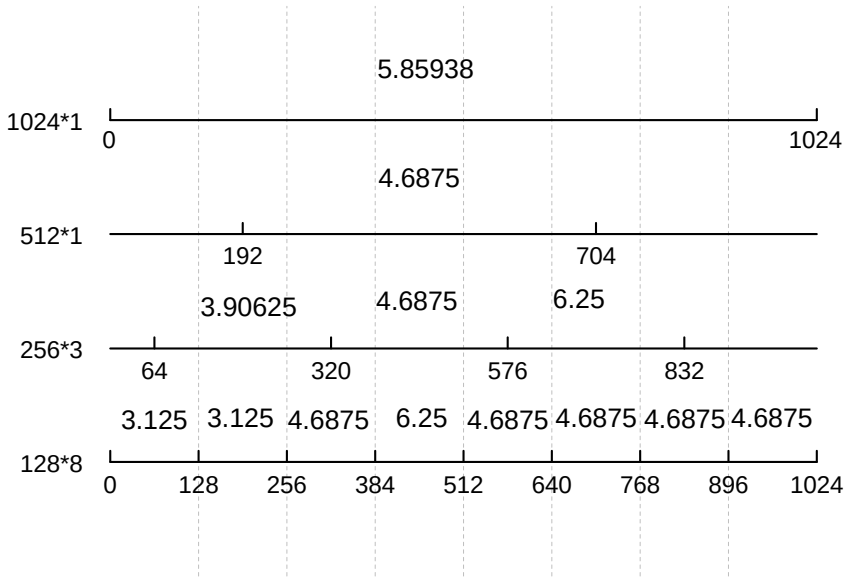


**Figure 3.6.** Peak Frequency Components with Different Windows

Due to this design, the peak frequency may have correlation among different window length. However, the above figure shows that since the STFT is calculated within different time period and using different window length, which leads to different frequency precision accuracy, the correlation among different window length is acceptable.

## 3.4 Networks Communication Module Design

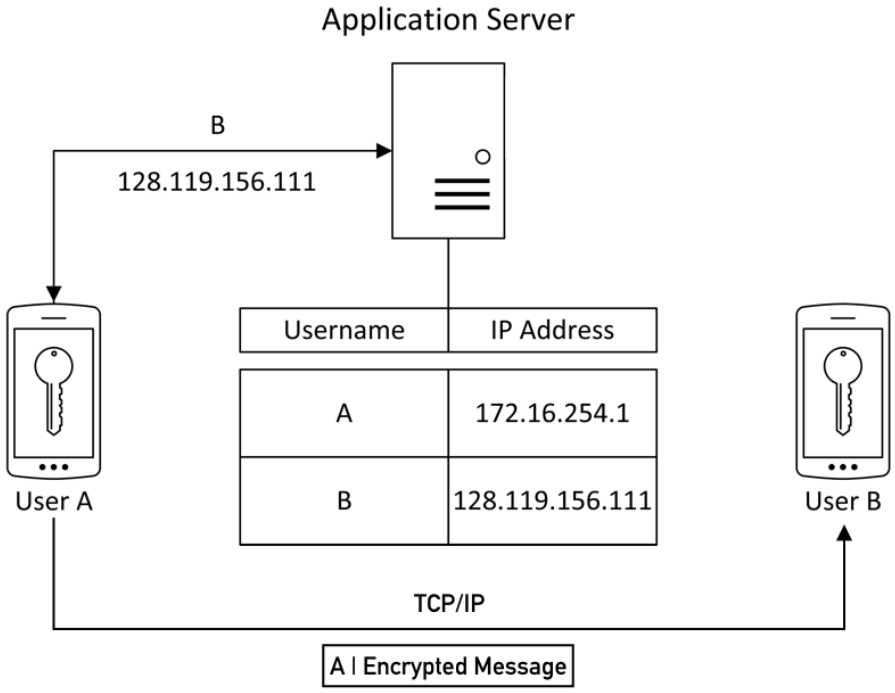


**Figure 3.7.** Networks Communication Module Design

When user A wants to send message to user B. User A should choose user B in its contact list and select Send Message option. After user A types in the content of message and confirms to send, this message will be encrypted with the corresponding symmetric key stored in local database. Meanwhile device A queries the application server about ip address of the destination device. Then send the encrypted message with sender username to the destination device directly under TCP/IP standard. The message will be displayed to user B after decryption.

This mechanism is kind of similar as Domain Name System (DNS). Though not complex, it is able to secure the messages sent between users as a trade off possibility of losing message. It obviously works for common situation, however, it may fail if user B does not update its IP address in time. Thus when before sending message, both the sender and the receiver have to log in their account to update their IP. Now

assume user B does not update IP and user A send message to its previous IP, the message may lost or even pushed to other application user. This is same scenario as an attacker intercepts and captures a encrypted message. Fortunate the content of message would not be revealed as a consequence of neither other users nor the attacker does not have the key to decrypt the message. Thus even user B will not get the message, it is secure. The same scenario may happen to one-time pad as well and it is still accepted as unbreakable cipher technique.

# CHAPTER 4

# PROTOTYPE IMPLEMENTATION

## 4.1 System Functionalities

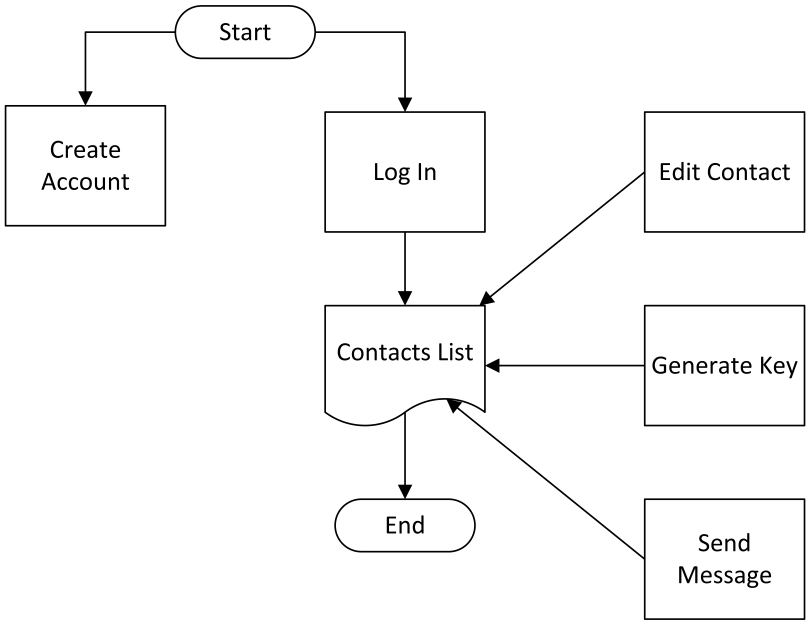


**Figure 4.1.** Shake Share Functionalities

As mentioned in section 3.1 System Architecture, this application is comprised of three component modules, which are database module, key generation module and networks communication module. Functionalities for each module will be illustrated in this chapter. Figure 4.1 is the functional flow block diagram of the application. It shows interrelationships among each module and guides the design of user interface. Start or End denote launch or terminate this single-threaded process. Contacts List is a user interface to display contacts' information fetched from local SQLite database

rather than an option. Create Account, Log-In and Edit Contact are related to database module since these functionalities are designed to manage the entries in both local SQLite database and the server database. Generated Key and Send Message are user operations for the other two modules.

## 4.2 Database Module Implementation

This mobile application has a local authentic SQLite database which encrypted with cryptographic protocols to store both information about users and their contacts. For local SQLite database, there is one table named LocalUsers for user log-in. Within this table, two variables which are username and password are stored for each user who has successfully created an account using this mobile device. Meanwhile, there is another table for each registered user named by the username stored in the database. Information about a certain user's contacts is stored in this table including contact names, alias, keys and message contents. At the application server database, a table named ServerUsers including username and IP address is stored working as the look-up table for DNS.

As a trade off for not relying on servers or any third party authorities, this mechanism has to trust devices to a great extent. Only username and its corresponding IP address are stored in the server. Thus even if the server is compromised, an attacker will obtain nothing but the IP address where users logged in their account last time. All the confidential data, especially the symmetric key used to encrypt and decrypt messages, is stored in the local database. Therefore, it is users responsibility to keep their device secure and not give out their database files. To guarantee this strategy performance, keep the secrets completely concealed is definitely required. In case of losing the device, the user should try to inform all his or her contacts at once to prevent leaking private information and unfortunately has to start over from the very beginning.

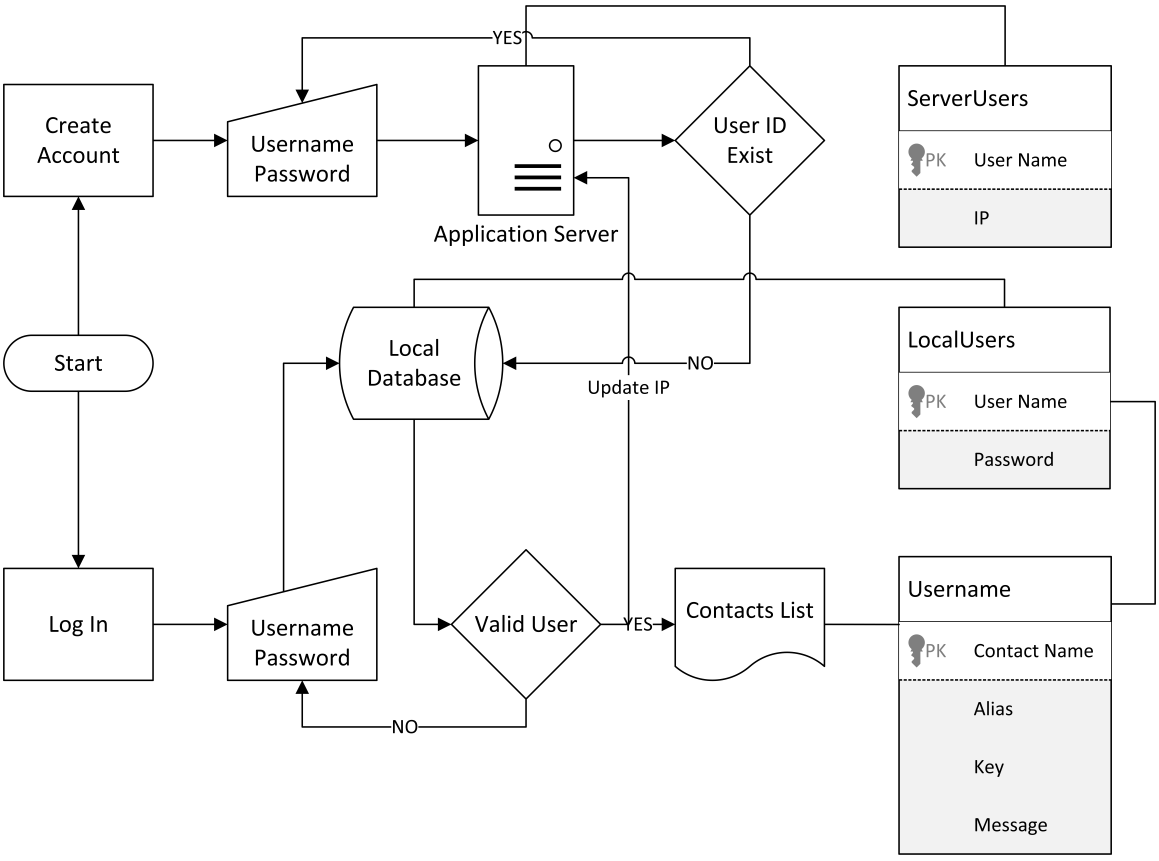


**Figure 4.2.** Create Account and Log In Flow Chart

Once the application is stated, secure authentication is required. Users are able to create their account with username and password or to log-in with their existing account information. Of course, log-in will not succeed unless creating an account first. To create an account, a user has to type in username and password then the application will send the username to the application server to check if this username has already existed in the system. Username must be unique in this mechanism. Otherwise, it may mess up network communication based on the principle of DNS. If the username is valid, the application server will store this username and corresponding IP in the ServerUsers table and send back an approval ticket for inserting this user information into LocalUsers table in the local database. To log-in, if the input username and password combination is valid, matches an entry stored in the

local database, the application will update IP address of this user's current device to the server.

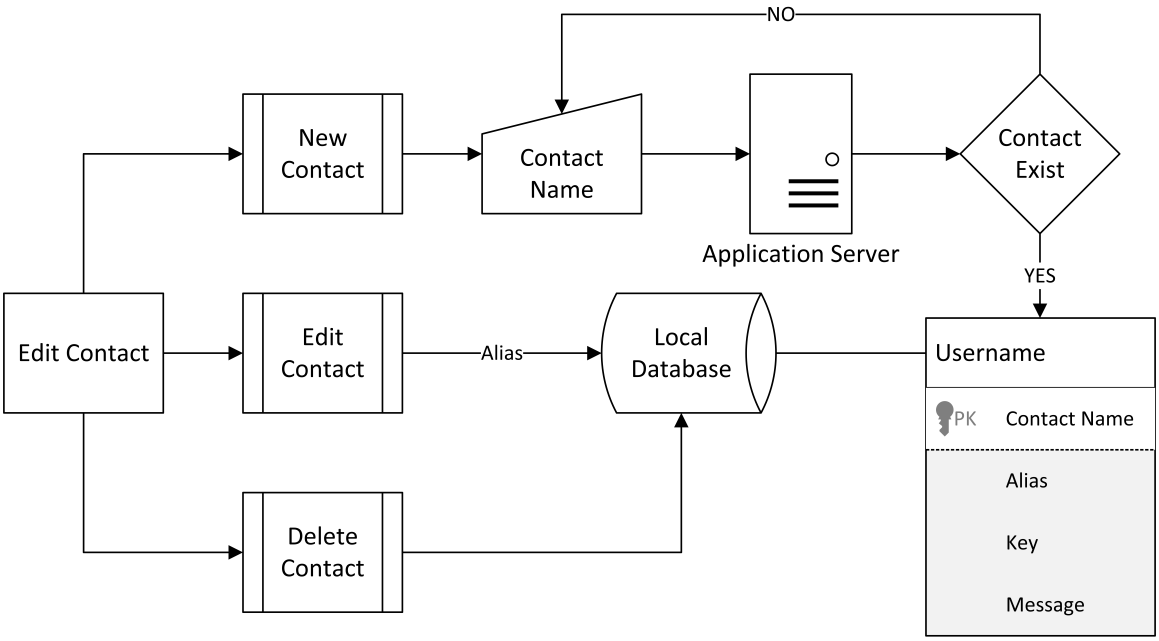


**Figure 4.3.** Edit Contact Flow Chart

After log-in successfully, the contacts information of a particular user will display on the screen. At this point, a user is able to add a new contact, delete or edit contacts' alias. Alias is the contact name shown on the screen to help the user identify every contact. To new a contact, user types in the contact name and the app send this contact name to the server. If the contact name is already registered in the system, the server will send back an approval ticket for inserting this new contact name into the table named by the username in the local database. Edit and delete contact are the SQL commands to manipulate the local database file.

## 4.3 Key Generation Module Implementation

Before a contact's name, there is a star shape icon. If this icon is grey, it means there is no key stored for this contact. Otherwise, if the icon is golden, it means

there is a key stored for this contact. To generate a cryptographic key with a certain contact, two participants have to log-in their account and add each other into their contact list first. Then for each participant, select the corresponding contact name and click on the generate key button to start key generation activity. Afterwards, hold two desired devices in a fixed position in one hand. Then start to shake the joint object randomly in 3 denominational space for about 5 seconds. Strong shake as a start point is required. The first ring tone means sensor data recording start, while the second one means the end of the recording. During this period, the user should keep shaking the involved devices. The algorithm of the key generation module will calculate the symmetric key only imposing sensor data of each device. An encrypted sentence will be displayed on the screen. If the two users got the same encrypted sentence, that means they got the exactly same key. The common secret will be updated into the local database after a user manually confirm the key is generated successfully.

### 4.3.1 Linear Acceleration Sensor

The devices that used to test the prototype implement is Nexus 7 (2012) equipped with both accelerometer and linear acceleration to sense device motion. The parameters for these two sensors are shown in the following table.

**Table 4.1.** Parameters for Acceleration Sensors in Nexus 7 (2012)

| | MPL* Accelerometer | Linear Acceleration Sensor |
|---|---|---|
| Manufacturer | InvenSense | Qualcomm |
| Version | 1 | 1 |
| Range | 39.226593 $m/s^2$ | 39.226593 $m/s^2$ |
| Resolution | 0.001190 $m/s^2$ | 0.001190 $m/s^2$ |
| Min Delay | 5000 $\mu$ s | 5000 $\mu$ s |
| Power Consumption | 0.500 mA | 4.100 mA |
| Accuracy | Unreliable | High |

* MPL - Motion Processing Library
Data is from Android applications - Sensor Kinetics Innoventions, MyDroid System Info and Android Sensors

Compared with accelerometer, linear acceleration sensor performances better in both synchronisation consistency and key strength. The reason is though MPL accelerometer consumes less power, linear acceleration sensor is able to eliminate interference of gravity and provides higher sensor accuracy. First, this approach is triggered when the shaking amplitude is larger than 25 $m/s^2$ as will be explained is the section 4.3.2 Sampling. However, accelerometer reads a magnitude of g of 9.8 $m/s^2$, thus it is much easier to pass the threshold value. In few trials adopting accelerometer, the device started recording even no shaking at all. Second, with the affect of gravity, even though shaken really hard, the FFT peak always falls in a narrow range of low frequency domain, that leads to reduce information entropy and therefore affects the accuracy of this approach.

### 4.3.2 Sampling

Starting recording simultaneously is crucial to the performance of this application, since devices are inherent unsynchronized. If the two involve devices record sensor data at different starting points of the shaking process, the samples that collected during the motion will have time displacement between the two devices. As a consequence, the key features will not be the same since they are actually extracted from two different acceleration data sets and the algorithm of extraction - STFT - is time sensitive. To make the devices start recording at the same point of the shaking process, the strategy is setting a threshold of sensor data magnitude as a trigger. Compared with other ideas, like measuring the proximity between two objects or giving a command by voice, this strategy is the most convenient and accurate, since there are no additional environmental or position sensors involved other than the motion sensor and the recording will start right away after a strong shake is detected. After test several magnitude values, I choose a threshold of 25 $m/s^2$ after dimensionality reduction as the implicit synchronization condition.

Sampling rate is very important as well. Usually, Android API SensorManager provides four fixed intervals at which sensor events are sent to applications. Those four sampling rates are shown as table below.

**Table 4.2.** Delay Interval Options for Android Sensors

| Option | Delay Time | Ideal Sampling Rate | Average Sampling Rate |
|---|---|---|---|
| SENSOR_DELAY_NORMAL | 200 milliseconds | 5 Hz | 4.89 Hz |
| SENSOR_DELAY_UI | 60 milliseconds | 16.667 Hz | 14.8945 Hz |
| SENSOR_DELAY_GAME | 20 milliseconds | 50 Hz | 49.6481 Hz |
| SENSOR_DELAY_FASTEST | 0 milliseconds | 200 Hz | 198.594 Hz |

[1] Delay Time refers to developer.android.com
[2] Average Sampling Rate is from Android application Accelerometer Frequency

In this work, the delay rate is set as SENSOR_DELAY_FASTEST, that means sampling rate is about 200 Hz. By doing that, even though consuming power, it guarantees the accuracy of results. Tests with other rates shown that none of those rates is fast enough for the desired performance. Another thing is time intervals between every two samples are not quite equivalent due to the imprecise sensor hardware. However, because we will transform the time domain data to the frequency domain, it will not affect the accuracy of this approach.

After the starting point, the sensor will collect 1024 data and then end recording automatically. Considering the sample rate is almost 200 Hz, it will last for around 5.12 seconds. Tests shown when set the totally collecting number of samples as 2048, that means the duration for collecting samples is around 10.24 seconds at 200 Hz, even though the two devices start recording at the same point, sometimes the recording phase end up at different time. It is because of time slicing in multi-threading is controlled by the Android operating system.

User confirmation is needed to verify that the two users already get the shared secret successfully without revealing it on the screens directly. Obviously, if the key is visible to users, it will benefit attackers a lot and make the entire system vulnerable.

However, the situation is not expected which is the two users think they already get the key but find out it does not work after they are physical separated. The solution is encrypting a certain sentence with the common secret using AES on each device, then displaying this encrypted sentence on the screen. If the two users get the same encrypted sentence that means they generate exactly the same key. After that, they need to confirm the key manually and say "I will talk to you later and take care.". The key will be updated according to the contact name in the database after being confirmed. Every time a new key is confirmed, it will update the old one if exist.

### 4.3.3 Key Sequence

As mentioned in section 3.3.5, the frequency range that considered in this approach is (2.73438 Hz, 8.78906 Hz). According to the equation of converting index i of FFT results to frequency f, $f = i * Fs/N$, get frequency information for each window length shown as the table below. Segments for each window length are as designed in figure 3.2.

**Table 4.3.** FFT Index to Frequency

| | Frequency Range | Increment | Numbers | Bits | Segments |
|---|---|---|---|---|---|
| 1024 | 2.73438 $\sim$ 8.78906 | 0.1953125 | 32 | 5 | 1 |
| 512 | 2.73438 $\sim$ 8.59375 | 0.390625 | 16 | 4 | 1 |
| 256 | 3.125 $\sim$ 8.59375 | 0.78125 | 8 | 3 | 3 |
| 128 | 3.125 $\sim$ 7.8125 | 1.5625 | 4 | 2 | 8 |

For each 5.12 sec, first convert decimal FFT results index to a binary string, that will be a $1 \times 5 + 1 \times 4 + 3 \times 3 + 8 \times 2 = 34 bits$ binary sequence. To ensure the security to a higher level, more than 200 bits are required. Therefore, users have to shake six times, which will last for $5.12 \times 6 = 30.72 sec$, to get a $34 \times 6 = 204 bis$ binary string. Then hash this binary string using MD5 to 128 bits for that it can be used as an AES key. Convert this string from binary to hexadecimal and store it in the local database.

## 4.4 Networks Communication Module Implementation

To send message, a symmetric key for encrypting messages is the prerequisite. If a common key is generated successfully and stored in the local database, a golden star icon will shown before the contact name. Otherwise, the star icon will be grey. Select the desired contact first, then type in the message contents and press send button. The message will be first updated in the local database and then encrypted using AES with the key, meanwhile the application will query its server about the IP address of the destination device. After get the response, IP address of the associated device, from the server, the sending device is able to deliver the encrypted message to the intended target device directly using socket under TCP/IP standard. After deciphering the encrypted message, the receiving device will insert the message content in its local database.

# CHAPTER 5

# EVALUATION AND DISCUSSION OF RESULTS

This chapter first presents the security model of the proposed system, including security requirements and attacker capabilities. Then based upon this security model, rules and procedures of testing experiments are set up. This chapter focuses on analyzing the experimental results and discussing the performance of the system. Currently, preliminary experiments of the proposed prototype show that this approach works well with a success rate of about 60%, a key with an average entropy of 63.9 bits are generated, in the meanwhile, it is able to separate shaking movements from others that are not shaken together with a 0 percent false negative rate.

## 5.1 Security Model

Security model, a formal statement of security policies, outlines how the necessary logic and rules are to be implemented and provides proofs for security evaluation [13].

### 5.1.1 Security Requirement

Security requirements describe the necessary schemes that should be developed to achieve a certain security. To ensure the security of this mechanism, the following 4 requirements must be fulfilled.

A Hold two devices tightly in a fixed position with one hand and shake them together should lead to a consistent key.

B The key characters should be true random considering they are captured from arbitrary movement pattern.

C Imitate a gesture without actually shaking device with the other one should fail to get the same key.

D Need to work if sensor not x,y,z aligned

### 5.1.2 Attacker Capabilities

The security of this approach is guaranteed based on the assumption that attacker capabilities are limited as following. Otherwise, it will case security vulnerabilities.

A Attacker can observe the shaking process but do not have access rights to take other devices.

B Attacker cannot read out others' sensor data.

## 5.2 Experiments Design

Based on the above security model, the experiments are first divided into 3 major groups as following.

A One person shake two devices together with one hand.

B One person shake two devices simultaneous with one device in each hand in the most similar movement pattern.

C Two person shaking two devices with one shaking a device in arbitrary pattern, at the same time, the other person try his best to imitate the movement pattern.

Ideally, one hand experiments should generate identical key successfully to prove this approach work, in the contract, two hands or two person experiments are expected to get different keys verifying the system is able to protect against potential attacks. As mentioned in 2.2, comparison between two trial samples that randomly pick from one hand experiments as long as they are not generated at the same time is not

sufficient to prove the capacity of resisting attacks. An attacker may imitating the motion based on the observation of a motion or even perform replay attack after video tap the motion using high-speed camera. Thus, two person experiments are necessary and two hands experiments are designed for higher standard.

Two factors may affect the performance, that are sensor 3-axes aligned, which each axis is repetitively in same direction, and shaking speed. So for each group, contrast experiments related to these factors must be conducted. First divide each group into 2 subgroups: sensor axes are aligned and non-aligned. And for non-aligned, 4 cases are concerned that are x and y axes are in opposite direction, x and z axes are in opposite direction, y and z axes are in opposite direction. For the speed of motion, both fast and slow will be tested. As discussed in section 4.3.2, for every experiment, an individual needs to shake a device or two devices for around 5.12 sec to get 34 bits key.

One hand experiments consist of 52 shaking experiments, including 40 sensor axes aligned tests and 12 non-aligned tests. 24 out of 40, that is 60% of totally tests, extracted symmetric key successfully when sensors are aligned while the results show that when sensors are not aligned, this application fails to work. Further discussions about how the sensor alignment affects the performance as well as how motion speed affects the performance are in section 5.5.

For two hands and two person experiments, 15 tests of each of them have been done. None of the totally 30 experiments leads to the same key, which satisfied the expectation.

## 5.3 Randomness of Keys

The randomness of key sequences are crucial to the security level. To examine it, all the generated key bits from the above experimental tests are passed into NIST randomness tests [14]. Unfortunately, the key sequences did not pass any of

the randomness tests. A possible reason is the sample scale, which right now is 70 experiments with 34 binary bits for each one, is not large enough.

In addition, the distributions of these key sequences, denoted in decimal, show that most of the experiment results fall in low frequency range due to people are more likely to perform a relatively slow shanking processes. The figures below are the histograms of key characters for each STFT widow length.

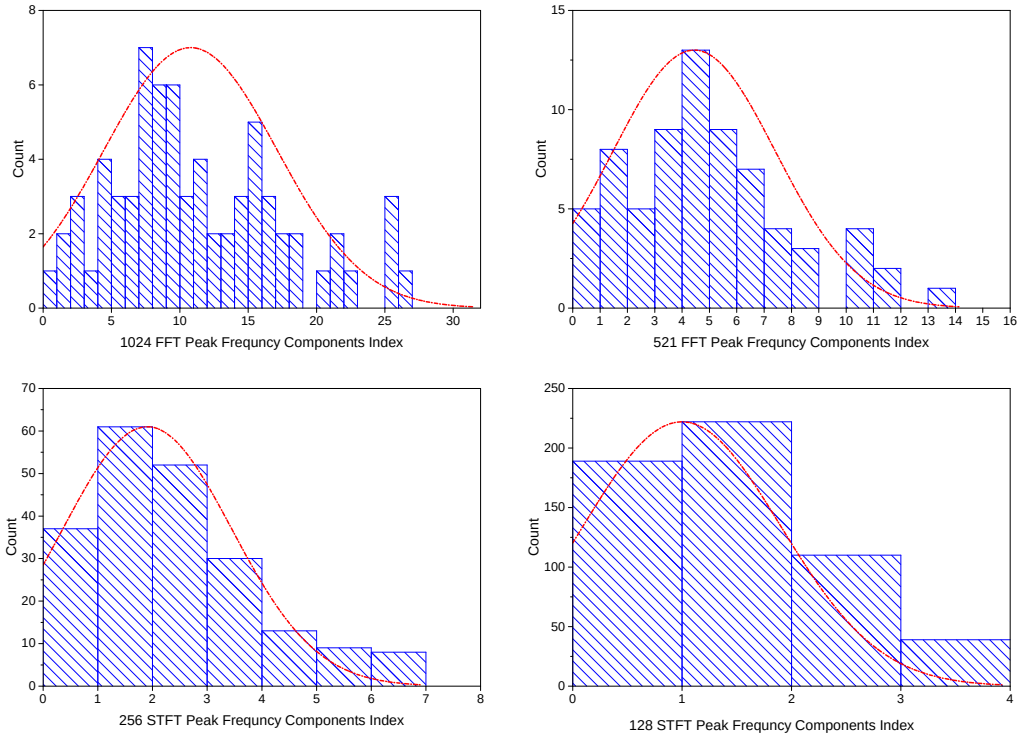


**Figure 5.1.** Distribution of Key Characters

Red lines are the normal distribution curve based on the histograms. They show that actually, the key characters distributions are left shift of normal distribution. Enlarge the experiment sets and indicate testers to shake in various speed may change the normality test result.

## 5.4 Difference between keys by Imitation

Two sequences of different shaking process should result in different keys. Moreover, the number of different key bits shall be as much as possible to enhance the security level. If only few bits are different, there maybe a chance for an attacker to do error correction and therefore get the same key. Check the different bits for each of the 30 two hands and two person experiments, the average number of different bis between two key sequences generated by imitation motion of two person is 7 bits. The more narrow window is, the more possibility to get different bits.

## 5.5 Performance Discussion

The exactly number for each kind of test and corresponding results are shown in the table below.

**Table 5.1.** Results of Experiments

| | Aligned | | Non-aligned | | | | | |
|---|---|---|---|---|---|---|---|---|
| Opposite axes | | | X & Y | | X & Z | | Y & Z | |
| Motion Speed | Fast | Slow | Fast | Slow | Fast | Slow | Fast | Slow |
| One Hand | 10/20 | 14/20 | 0/2 | 0/2 | 0/2 | 0/2 | 0/2 | 0/2 |
| Two Hands | 0/8 | 0/8 | 0/2 | 0/2 | 0/2 | 0/2 | 0/2 | 0/2 |
| Two Person | 0/8 | 0/8 | 0/2 | 0/2 | 0/2 | 0/2 | 0/2 | 0/2 |

How related factors that may affect the performance are discussed based on the above table.

### 5.5.1 Sensor Alignment

In theory, as applied and proved, the 3 axes reduction should work. However, in practical, it does not work. The reason maybe the quality of hardware. Pick 3 test samples form two hands experiment, I found that Y axe has most significant affect on the performance, that is when Y and Z axes are in the opposite direction, the sensor data are not as matching as when the 3 axes are aligned.

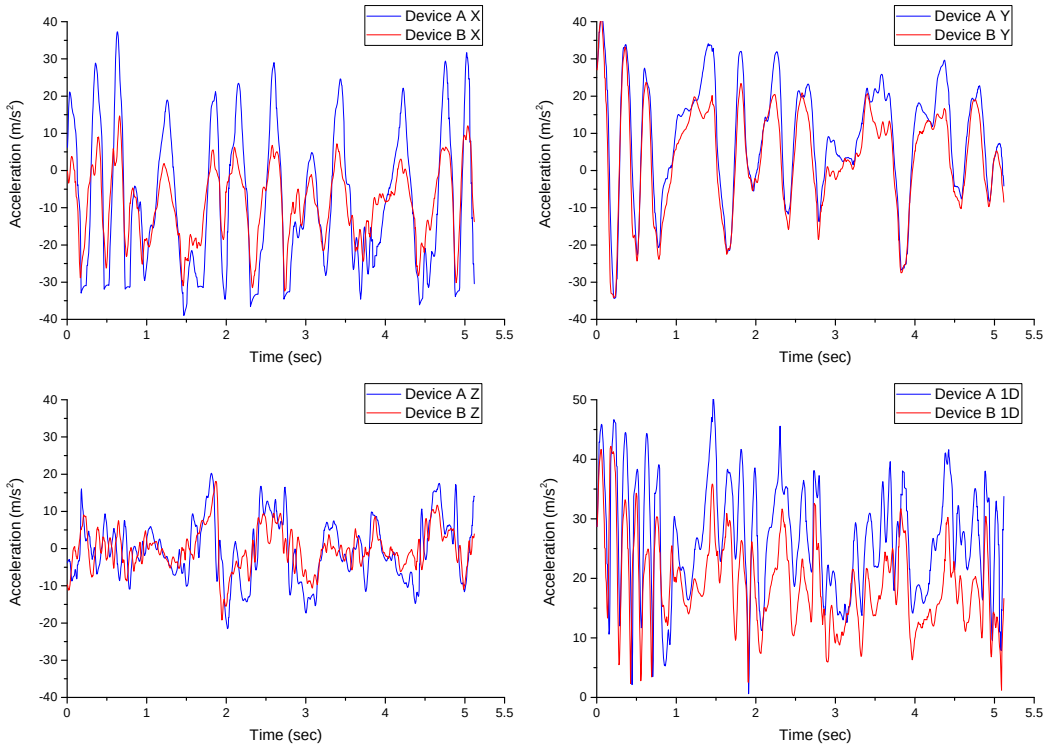


**Figure 5.2.** Time Domain Signal When Y and Z Axes Are In the Opposite Direction

### 5.5.2 Movement Speed

Sensor generates random bits at all time, and those random bits have affect on high speed motion performance. So the accuracy is better when users shake the devices in relatively slow shaking process. By relatively slow, it means the frequency is under 5 Hz. However, I believe that with higher precision sensors, the performance shall be improved.

### 5.5.3 Movement Pattern

In general, most of shaking processes are fast up and down movements along three axes, may combined relatively few with rotations [12]. Results show that the approach performs better for up and down movements than rotations. There is a certain possibility that the approach fails when the movement involved a large extent

rotations. The reason is the adopted sensor - linear acceleration sensor measures acceleration forces not rotational forces. To enhance the performance, gyroscopes or rotational vector sensors shall be introduced into the system.

# CHAPTER 6

# CONCLUSION

Pairing mobile devices by shaking is an easy, fast and secure mechanism, given that the keys are randomly generated and invisible and no data exchange takes place between devices or with a server.

Compared with previous works that only deal with acceleration data in time domain or frequency domain, extracting identical key features using STFT, which is the basic method of joint time-frequency analysis, not only is feasible but also improves the accuracy to a great extent.

Practically, this approach has good performance with an accuracy of 60% and key entropy of approximately 63.9 bits for extracting a symmetric key from two mobile devices independently of each other from a shared motion pattern. In the meanwhile, it is able to distinct the shaking processes that do not take place together. Therefore, security level is risen, and better protection is provided.

Additionally, with a particular networks protocol, symmetric key generation is able to guard user sensitive information against eavesdropping or man-in-the-middle attack.

# BIBLIOGRAPHY


[1] P. S. Lab. (2014) Infographic: The golden age of mobile. [Online]. Available: http://pennystocks.la/blog/golden-age-of-mobile

[2] S. Oriyano and J. Doherty, *Wireless and Mobile Device Security*, ser. Jones & Bartlett Learning information systems security & assurance series. Jones & Bartlett Learning, LLC, 2014. [Online]. Available: https://books.google.com/books?id=Oa3koAEACAAJ

[3] D. Bosomworth. (2015) Mobile marketing statistics compilation. [Online]. Available: http://www.smartinsights.com/mobile-marketing/mobile-marketing-analytics/mobile-marketing-statistics/

[4] V. R. (2014) 7.7 billion mobile devices among 7.1 billion world population by the end of 2014. [Online]. Available: http://dazeinfo.com/2014/04/29/7-7-billion-mobile-devices-among-7-1-billion-world-population-end-2014/

[5] Z. D. Boren. (2014) There are officially more mobile devices than people in the world. [Online]. Available: http://www.independent.co.uk/life-style/gadgets-and-tech/news/there-are-officially-more-mobile-devices-than-people-in-the-world-9780518.html

[6] Cisco. (2016) Cisco visual networking index: Global mobile data traffic forecast updates 2015-2020 white paper. [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html

[7] Wikipedia. Bump(application). [Online]. Available: https://en.wikipedia.org/wiki/Bump_(application)

[8] C. Arthur. (2013) Google buys smartphone file-swapping app bump as sharing grows. [Online]. Available: https://www.theguardian.com/technology/2013/sep/17/google-bump-smartphone-apple-airdrop

[9] S. Anywhere. Send anywhere. [Online]. Available: https://send-anywhere.com/

[10] H. Yuzuguzel, J. Niemi, S. Kiranyaz, M. Gabbouj, and T. Heinz, "Shakeme: Key generation from shared motion," in *Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), 2015 IEEE International Conference on*, Oct 2015, pp. 2130–2133.

[11] R. Mayrhofer and H. Gellersen, "Shake well before use: Intuitive and secure pairing of mobile devices," *IEEE Transactions on Mobile Computing*, vol. 8, no. 6, pp. 792–806, June 2009.

[12] D. Bichler, G. Stromberg, M. Huemer, and M. Löw, *UbiComp 2007: Ubiquitous Computing: 9th International Conference, UbiComp 2007, Innsbruck, Austria, September 16-19, 2007. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, ch. Key Generation Based on Acceleration Data of Shaking Processes, pp. 304–317. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-74853-3_18

[13] S. Harris, *CISSP Practice Exams*, 2nd ed. McGraw-Hill Education Group, 2012.

[14] N. I. of Standards and Technology. (2000) The nist statistical test suite. [Online]. Available: http://csrc.nist.gov/groups/ST/toolkit/rng/documentation_software.html