

Using User-aided Authentication to Secure an Ad-hoc and Ubiquitous Network

Graduate School of Systems and Information Engineering

University of Tsukuba

November 2014

Oyuntungalag Chagnaadorj

Abstract

The increased popularity of mobile devices, such as laptops, mobile phones, tablets, accessory devices, and more, brings new challenges in the area of network security. In ubiquitous environment, where ad-hoc networks may be formed using mobile devices, virtual connection or pairing is required before transferring any data wirelessly. Large numbers of researches have focused on the user-aided authentication to establish a secure connection between mobile devices, commonly referred as *secure device pairing*. The user-aided authentication is an authentication technique that is generally based on a human perceivable channel, called out-of-band (OOB) channel, through which the authentication data can be transferred.

A significant number of OOB channels have been proposed so far. The principle limitation of these approaches, however, is that every one of them is devoted to one particular situation and is not intended to be used for various mobile devices. In general, a universally applicable OOB channel is unlikely to exist in the near future due to the different capabilities of a wide variety of mobile devices as well as human users who have diverse preferences. Therefore, if a mobile device possesses as many as possible OOB channels, its possibility of successful pairing in ad-hoc and ubiquitous environment will increase. Based on this concept, we have defined our research vision, “the more OOB channels, the more security”. To support the vision, we have proposed two new methods.

First, we have introduced a new OOB channel that uses an accelerometer-based gesture input. Gesture-based OOB channel is suitable for all-kinds of mobile devices, including input/output constrained devices, as the accelerometer is small and incurs only a small computational overhead. In our OOB channel, one device converts the authentication data into a sequence of gestures and informs a user of them. The user then performs the gestures one-by-one with the second device that has an embedded accelerometer to transfer the data into it. We have implemented a prototype system and conducted thorough usability analysis to compare our gesture-based OOB channel with the existing OOB channels. Result showed that the gesture-based OOB channel has reliable performance with a mean completion time of 16.86 seconds with SD=4.2 seconds, as well as a mean error rate of 0.13 per transfer with SD = 0.34. Four different gesture sets were used in this experiment to determine which library size would be more usable in practice. The result suggested that library size of 8 gestures was the most effective among our tested sizes. We have also conducted another

experiment to determine whether user-defined gesture templates improve the accuracy of gesture recognition. Contrary to expectations, this study did not find a significant difference between the predefined and user-defined templates in terms of the error rate.

Second, we have designed a new secure group association method that utilizes various types of OOB channels. The term *secure group association* is generally understood to mean that every device pairs with other group members. There has been little discussion of setting up secure connections among a group of mobile devices. Some security protocols have been proposed, but they have tended to focus on a scenario whereby all devices must be physically located in the same place to perform the association. In contrast, our method is designed for a broader range of scenario and does not require all devices to be in one place at one time. It occurs in an accumulative manner, allowing each device to join the group independently from the other group members and their associations. To join a group, a device must perform secure pairing with one of the group members. Digital certificates are then issued upon the successful authentication. Once certificates are exchanged between the new device and the member, pairing with other group members becomes automatic owing to the certification path. We have implemented a prototype system and conducted a comparative user experiment in order to ensure viability of the proposed method. The result proved that the accumulative group association spends the same amount time as the simultaneous group association. In addition, we have reported comparison on accumulative and simultaneous associations and it clearly demonstrates our proposed method has a number of attractive features, such as stronger scalability, greater flexibility, unlimited group size, better device diversity and etc, over the existing protocols.

In summary, there will be more portable devices and more collaborative work in the near future, and the user-aided authentication will still play an important role to secure ad-hoc and ubiquitous networks. We hope that our proposed methods contribute to build the secure environment, where the security is always available on demand.

Acknowledgments

I am heartily thankful to Professor Jiro Tanaka, my thesis supervisor, for his many valuable suggestions, precise directions, and kind encouragement. I also greatly thank Assistant Professor Simona Vasilache for her English proofs, generous suggestions, and kind support. I also thank Associate Professor Shin Takahashi, Associate Professor Kazuo Misue, Associate Professor Buntarou Shizuki for their valuable advice and suggestions regarding this research. I would like to thank the committee faculties, Professor Eiji Okamoto, Professor Kazuhiko Kato, Associate Professor Akira Sato, and Associate Professor Hirotake Abe, for their many useful comments on this research. I am also grateful to all the members of the IPLAB, Interactive Programming Laboratory, University of Tsukuba, for giving me many opportunities to discuss my research with them and helping me in my experiments.

I would like to thank my mother, Dorjpurev Baldan, for her continuous support and encouragement. I extend my sincere gratitude to the soul of my father, Chagnaadorj Dondov for all his efforts and wishes. Finally, I like to thank my beautiful daughter, Egshiglen Bayasgalan, for making my life brighter and happier.

Contents

Abstract	ii
Acknowledgments	iv
List of Tables	3
List of Figures	4
1 Introduction	6
1.1 User-Aided Authentication	7
1.1.1 Authentication Protocols	8
1.1.2 Out-of-band Channels and Their Classification	9
1.2 Motivation and Research Vision	10
1.3 Dissertation organization	11
2 Gesture-based Out-of-band Channel	12
2.1 Motivation	12
2.2 Related work	13
2.2.1 Out-of-band channels	13
2.2.2 Accelerometer-based Gesture Input	15
2.3 Proposed Gesture-based OOB Channel	16
2.4 Prototype System	18
2.4.1 Gesture Library	19
2.4.2 Gesture Encoding	19
2.4.3 Gesture Recognition	21
2.4.4 Dynamic Time Wrapping Algorithm	22
2.4.5 User Interface	24
2.5 Experiment One: Usability Analysis	26
2.5.1 Purpose	26
2.5.2 Participants and Task	27
2.5.3 Completion Time	28
2.5.4 Error rate	29
2.5.5 Learnability	30
2.5.6 Subjective Ratings	31

2.6	Experiment Two: User-defined Templates	32
2.6.1	Purpose	32
2.6.2	Result	32
2.7	Discussion	33
2.8	Summary	34
3	Accumulative Secure Group Association	36
3.1	Motivation	36
3.2	Related Work	37
3.2.1	Secure Group Association Protocols	37
3.2.2	Group Association	39
3.3	Secure Group Association Types	40
3.4	Proposed Accumulative Secure Group Association Method	42
3.4.1	Base Concept	42
3.4.2	Centralized Model	44
3.4.3	Certification Path Pairing (CPP) Protocol	46
3.4.4	Role of the groupHub	48
3.5	Prototype System	49
3.5.1	System Setup	49
3.5.2	Proposed Method Implementation	50
3.5.3	SAS-GMA Protocol Implementation	53
3.5.4	User Interface	55
3.6	Comparative User Experiment	56
3.6.1	Purpose	56
3.6.2	Tasks	57
3.6.3	Participants	59
3.6.4	Result	60
3.7	Discussion	61
3.8	Summary	63
4	Conclusion	64
4.1	Contributions	64
4.2	Future Work	66
	Bibliography	67
	List of Publications	78

List of Tables

2.1	List of selected gestures	20
2.2	Summary of proposed interfaces	20
2.3	Gesture conversion sample of interfaces	21
2.4	DTW method: Constructing the distance matrix from two sequences	24
2.5	DTW method: Estimating the similarity cost	24
2.6	User Experiment: Mean completion time	28
2.7	User experiment: Paired tTest results	28
2.8	User experiment: Completion time of rounds	30
3.1	Summary of the existing SGA protocols	38
3.2	Characteristics of all-at once and one-by-one associations	40
3.3	Technical specification of handsets	50
3.4	Computational overhead of cryptographic operations	50
3.5	User experiment: Completion time - multiple users	60
3.6	User experiment: Completion time - single user	60
3.7	User experiment: Comparison between multiple and single user associations	61

List of Figures

1.1	Simple user-aided authentication	7
1.2	SAS (Short Authenticated String) protocol	9
2.1	Bluetooth Pairing: (a) Numeric comparison, (b) Passkey entry	14
2.2	Modified SAS protocol	17
2.3	Authentication process in the gesture-based OOB channel	18
2.4	Prototype setup of the user experiment	19
2.5	Gesture encoding: Depth6 conversion	21
2.6	User interface of experiment one: (a)the requester side - desktop PC (b)the verifier side - iPhone handset	25
2.7	User interface of experiment two: (a) evaluation (b) user list	26
2.8	Phone holding in the experiment	27
2.9	User experiment: Time distributions for a gesture input	29
2.10	User experiment: Mean error rates	30
2.11	Result of users' impression on the gesture input: (a) easiness (b) satisfactory of time (c) sureness of next successful try	31
3.1	Common multiple pairing scenario	42
3.2	The base concept of the accumulative SGA method	43
3.3	Limitations in the accumulative SGA method	44
3.4	Centralized model of the accumulative SGA method	45
3.5	CPP protocol in general	46
3.6	CPP protocol sequence	47
3.7	Sequence of the modified SAS protocol in the prototype	51
3.8	Sequence of the SAS-GMA protocol in the prototype	53
3.9	User interface of the proposed method: (a) experiment (b) settings (c) Key-store reading	55
3.10	User interface of the SAS-GMA protocol: (a) experiment (b) settings (c) barcode transfer	56
3.11	Task sequence of the proposed method: Completion time is measured from starting the first member pairing with the groupHub to completing CPP protocol of the last member.	57

3.12	Task sequence of the SAS-GMA protocol: Completion time is measured from launching association on all devices to ending the last device displaying the result of the verification.	58
3.13	User experiment: (a) multiple user - 5 participants are discussing a task before the association (b) single user - a participant is associating 4 devices	59

Chapter 1

Introduction

Over the last few decades there has been tremendous growth in the area of ubiquitous computing, and numerous portable devices that perform many complex operations and can be used in a wide variety of applications have been developed. Technologies including Wi-Fi, Bluetooth, and ZigBee have been created to enable wireless communication between these devices. However, compared to wired connections, wireless networks are more vulnerable to security threats, in particular eavesdropping and alteration, also termed man-in-the-middle (MitM) attack [1]. It is generally assumed that major security issues, including MitM, can be addressed if one's cryptographic public key can be authenticated. Authentication methods, such as using a pre-installed key or a trusted third party authentication, cannot be adopted, because wireless networks are usually set up on an ad-hoc basis involving unfamiliar devices. Therefore, a new authentication mechanism is needed for ad-hoc and ubiquitous networks. Large numbers of researches have been focused on the user-aided authentication to bootstrap the problem.

Throughout this dissertation, the term *secure device pairing* is used in its broadest sense to refer to establishment of a secure connection between two mobile devices. Similarly, the term *secure group association* is used to mean establishment of secure connections among a group of mobile devices. Both secure device pairing and secure group association have been employing the user-aided authentication to enable security.

There are many everyday scenarios where two or more mobile devices need to interact in a secure manner. The common secure pairing case occurs when two devices, such as a Bluetooth headset and a cell phone, or a PDA and a wireless printer, transfer sensitive data to and from each other. Likewise, a wireless body area network (BAN) with portable sensor

devices enables continuous monitoring of patients in the hospital. BAN is an emerging group collaborative case in recent years, and also deals with important medical information of a patient that needs more requirements for security [2].

1.1 User-Aided Authentication

The main principle of the user-aided authentication is to use human user's involvement in the authentication process. In this paradigm, an additional auxiliary channel that is perceivable by the user, called the out-of-band (OOB) channel, exists between two mobile devices, as well as the ordinary wireless channel, as illustrated in Figure 1.1. In this dissertation, *the requester* refers to the device that is about to be authenticated and *the verifier* refers to the device that does the authenticating. Various pairing protocols have been proposed so far, and Figure 1.1 depicts a simple one to elucidate how the user-aided authentication works. To authenticate the requester, the verifier receives both the cryptographic public key through the wireless channel and the hash of the same key via the OOB channel. The user involvement is required in OOB channel transmission and the data, transferred through OOB channel, is called *the authentication data*. The verifier then generates another hash for the requesters public key and checks it against the received hash data. If cross authentication is required, the same process is repeated in the reverse direction.

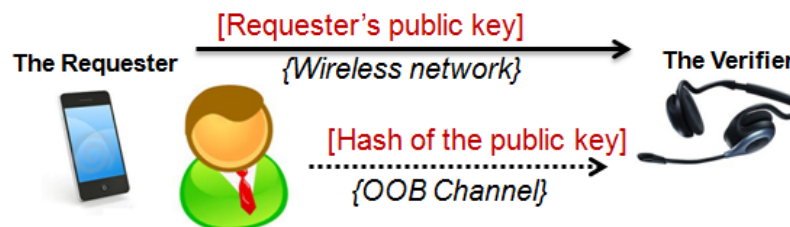


Figure 1.1: Simple user-aided authentication

One important question that needs to be answered before going any further, however, is whether the user-aided authentication can be trusted. Authentication in Public Key Infrastructure (PKI) is based on the concept that all entities have confidence in centralized Certificate Authorities and their issued digital certificates [3]. Similarly, the user-aided authentication assumes that a human user who is trying to pair mobile devices is trustworthy

and his or her action is expected to be sincere and honest. This assumption is quite normal because the user is the one who gets the benefits from the paired devices.

As it can be seen from Figure 1.1, the user-aided authentication consists of two independent yet inseparable parts: an authentication protocol and an OOB channel. The following subsections will give a brief overview of each one of them.

1.1.1 Authentication Protocols

The concept of the user-aided authentication was the first introduced in 1999, and at the same time the first protocol to form a secure wireless network was presented [1]. The new approach was then studied and refined further by many researchers, and as a result, various secure device pairing and secure group association protocols have been proposed [4–13]. Some of these protocols require a specific OOB channel whereas others are designed to operate on various mobile devices. Vaudenay et al. [14] have introduced a SAS (Short Authenticated String) protocol, which is well suited for many low-bandwidth OOB channels. It reduces the length of the authentication data to 15 bits while providing a reasonable level of security. Thus, the protocol has become favorable for many mobile devices that are not able to transfer large data because of their limited physical capabilities. It has also played an important role in our research.

SAS protocol is depicted in Figure 1.2. It is based on a cryptography function, called Commitment Scheme. It has several important features that make the scheme useful for a number of network applications [15]. Commitment Scheme consists of two functions: *commit* that is used to commit a value to keep it hidden to others and *open* that is used to reveal the committed value. In SAS protocol, both the requester and the verifier first choose random 15 bit data (Ran_R and Ran_V) independently. The requester device then commits on the chosen data together with its public key (PK_R). After reception of the requester's public key and the commit value c , the verifier device gives its own random data to the requester. Following this, the requester opens its commitment by sending value d , and consequently, the verifier can obtain the requester's random value without transferring it through insecure wireless channel. Finally, the requester computes SAS data ($Ran_R \oplus Ran_V$) and sends it through an OOB channel. The verifier then checks the received SAS data with its own computed one. If the requester's public key as well as other transferred values are tampered in transmission, the SAS authentication data on both side do not match.

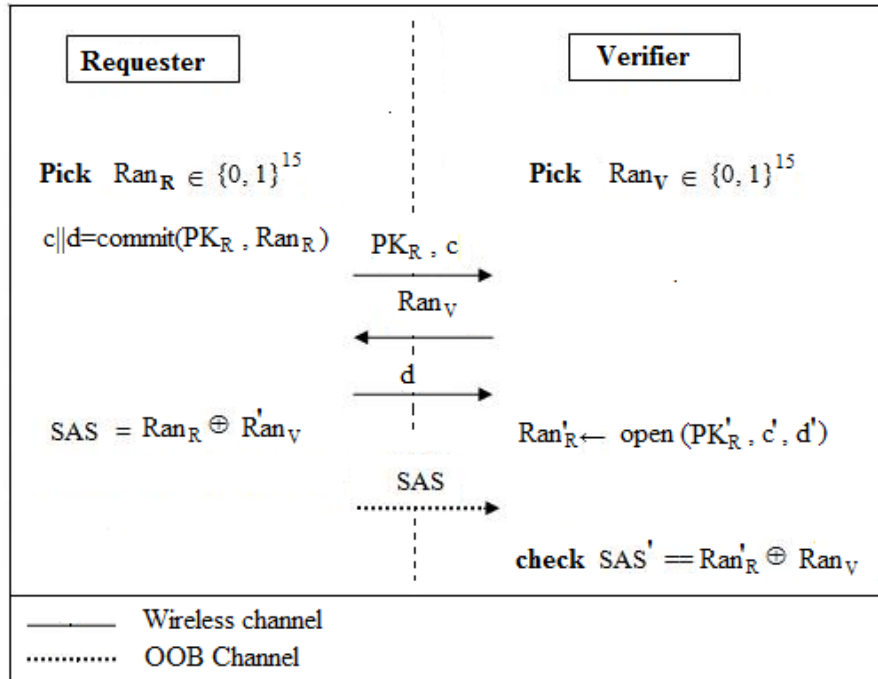


Figure 1.2: SAS (Short Authenticated String) protocol

Therefore, upon the successful completion, the verifier device can ensure that the received public key belongs to the requester device.

1.1.2 Out-of-band Channels and Their Classification

A significant number of OOB channels have been investigated in research. These OOB channels vary according to the type of physical interfaces they need. For example, they require mobile devices to possess different hardwares, such as a display [16–22], a camera [19, 23–25], a LED light ([23–26]), a speaker [26–30], a microphone [18, 26–28, 30], a keyboard [22], a button [28], a radio transceiver [31], a vibration [28], a ultrasound transceiver [32], a biometric sensor [33] and even a laser transceiver [34] to transmit the authentication data between them. This may explain why no single OOB channel has become a standard.

Many different classifications depending on various aspects of the user-aided authentication have been defined in recent studies [35–37]. However, in this dissertation, we emphasize the one that is based on the human user involvement. OOB channels may be divided into three main categories: device-to-device (d2d), device-to-human plus human-to-device

(d2h+h2d), and two device-to-human (2xd2h). In d2d OOB channels, the authentication data is transferred between two devices with the help of the user while in d2h+h2d OOB channels, the user receives the authentication data from one device and sends it further to another device. In both d2d and d2h+h2d, the device decides the outcome of the pairing. In 2xd2h OOB channels, the user himself compares the authentication data on both devices and decides the pairing outcome.

To elucidate how the authentication data is transferred via each type of OOB channel, we will look at three of them briefly. McCune et al.[19] proposed a barcode-based OOB channel (d2d). In this system, the requester encodes the authentication data into a two-dimensional barcode and shows it on its screen. The verifier reads the barcode using a camera. Goodrich et al.[18] developed a speaker display-based OOB channel (2xd2h). Authentication data are converted into text; one device speaks to it and the other shows the text on its screen. The user compares the texts to complete the authentication. Soriente et al.[28] introduced a button-based OOB channel (d2h+h2d). When the requester transmits the authentication data in the form of either beeps or blinks from an LED light, the user presses the button synchronously on the verifier side.

1.2 Motivation and Research Vision

Perhaps the most serious disadvantage of the user-aided authentication is that there are a large number of OOB channels available, however, none of them is generally accepted as a standard. And, none is likely to be in the near future because they currently cannot be adapted to the wide range of mobile devices that exist. A barcode-based OOB channel, for instance, requires a display for the requester and a camera for the verifier, which is not suitable for many input/output constrained devices. There is high likelihood of arising a problem, *unknown OOB channel*, when two devices pair spontaneously in the ad-hoc and ubiquitous networks.

Consequently, the problem leads to the conclusion that the adoption of OOB channels is not exclusively limited to using one OOB channel. A mobile device can simultaneously employ multiple OOB channels. Given this, our research vision, **“the more OOB channels, the more security,”** is formulated as follows. *Instead of trying to adjust one OOB channel to all mobile devices, which is hard to achieve, study possibilities that a mobile device uses as many as possible OOB channels.*

Interesting studies that support our research vision have recently been reported by Chong et al. [38] and Ion et al. [39]. The major findings of these researches reveal that there is no dominant device pairing method in the real life. Instead, people prefer different methods, and tend to select different OOB channels depending on the situation.

In order to support our research vision, we have proposed two new methods. First, we have introduced a completely new OOB channel that transfers the authentication data using an accelerometer-based gesture input. Second, we have designed a novel secure group association method, in which mobile devices can employ their desired OOB channels. Whereas, the existing secure group association protocols require all group members to use the same OOB channels.

1.3 Dissertation organization

The presented dissertation is organized as follows. Chapter 2 introduces our new OOB channel and reports conducted user experiments. Chapter 3 explains our proposed secure group association method and compares it with the existing protocols. Finally, chapter 4 concludes the dissertation and presents future work.

Chapter 2

Gesture-based Out-of-band Channel

2.1 Motivation

To support our research vision, “the more OOB channels, the more security”, we have introduced a new OOB channel that utilizes an accelerometer-based gesture input. We imply that the possibility of secure pairing will increase to some extent, if a mobile device possesses multiple OOB channels. For instance, assume that a wireless headset is able to pair using button-based OOB channel [28]. This channel requires the requester to have a LED light that may not be available for some devices. If the headset is also capable of pairing with a gesture-based OOB channel, it can pair with mobile devices that have either the LED light or a display.

There are a number of important reasons why we have chosen to consider the gesture-based OOB channel.

First of all, the gesture-based OOB channel is well suited for all mobile devices, including input/output constraint devices, because modern accelerometers are very small and lightweight. The accelerometer inside an iPhone 4S handset, for example, is $3mm \times 3mm \times 1mm$ thick and weighs 30 mg. In fact, accelerometers are already embedded in some commercial products, such as smart phones and Nintendo Wiimote.

Second, gestures represent one of the most promising interactive interfaces in the ubiquitous environment [40]; they are natural and intuitive for humans, and so gesture-based

operations are easy to implement and require little effort. Furthermore, most of the work on gesture recognition has been based on computer vision techniques [41]. However, gesture recognition from the tri-axis accelerometer data is an emerging technique for gesture based interactions. It requires a relatively small computational overhead compared to vision-based approaches.

Finally, there is an important and practical problem that related to the human user behavior, called *rushing user* [36]. A rushing user is a user who, in rush to connect her two devices, tends to skip through the pairing process, if possible. It has been shown that computer users are likely to be *task focused* [42]. For example, when a user logs on to the website of her bank, her focus is to pay a bill; she would tend to ignore any warning indicating a phishing attempt. Similarly, in the context of secure device pairing, when a user wants to connect her Bluetooth cell phone with a handset, she is eager to speak to someone. Such a rushing user behavior is quite common in practice. The findings of Kumar et al. [36] have revealed that d2h+h2d type of OOB channels are the most resistant to rushing user behavior whereas 2xd2h type of OOB channels are the most vulnerable against it. In d2h+h2d channels, the user is somewhat forced to perform the pairing process correctly, because otherwise he or she will not be able to connect the devices. Therefore, from the security point of view, d2h+h2d type of OOB channels, including our gesture-based channel, are preferable.

2.2 Related work

2.2.1 Out-of-band channels

A number of surveys of secure device pairing methods have been presented [35–37, 43]. They surveyed various aspects of secure device pairing techniques, such as the required hardwares, the user actions in the pairing process, protocol mechanisms, and factors that could influence usability of the pairing.

The success of user-aided authentication is dependent on the performance and effectiveness of the OOB channel, which basically depends on the user performing certain tasks correctly. Thus, comparative studies of existing OOB channels in terms of usability were independently conducted by Kumar et al. [36], Kaında et al. [44], and Kobsa et al. [45]. Findings in these studies revealed that in general, OOB channels offer varying degrees

of usability. Moreover, studies suggested that, among their tested techniques, traditional methods such as comparing or typing short strings into mobile devices yields lower error, and are the most robust, the fastest and the user's most preferred methods despite the fact that they require rich user interfaces.

Bluetooth connection is currently the most well-known representative of the secure device pairing in real life. Its authentication uses three different OOB channels [46] depending on the hardware capabilities of mobile devices: 1) If both devices have displays and are capable of entering yes or no, such as a laptop and a cell phone, pairing uses numeric comparison as shown in Figure 2.1 (a). In numeric comparison based OOB channel, a user compares the numbers in both devices. If they agree, the user enters acceptance in both devices, or otherwise the user enters rejection and the system cancels the pairing procedure. 2) If one device has an input capability but does not have a display and the other device has an output capability, such as a laptop and a keyboard, pairing uses passkey entry, as shown in Figure 2.1 (b). In passkey entry OOB channel, a user first reads a passkey displayed on a device and then enters the same key into other device. 3) In other cases, like at least one device does not have a display nor an input capability, such as a cell phone and a headset, pairing just connects the devices without using any OOB channel.

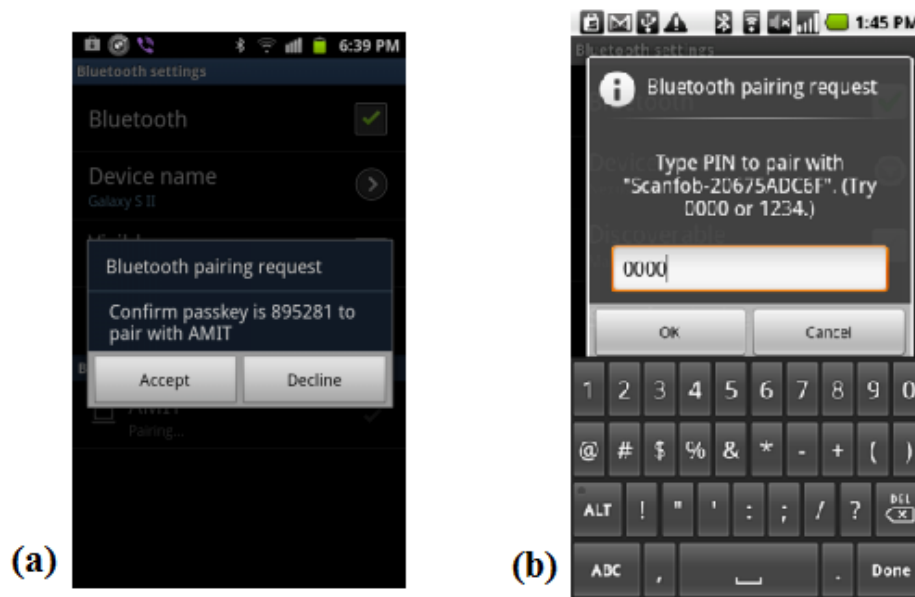


Figure 2.1: Bluetooth Pairing: (a) Numeric comparison, (b) Passkey entry

Another secure device pairing approach is to generate the same shared key in both devices without transferring any data through secure or insecure channels. Up to recently, few reports in this area exist. Mayrhover et al. [47] proposed a system whereby two mobile devices are held and shaken together to create a shared secret key. In this case, both devices must have embedded accelerometers, and the acceleration data are converted into the same key. Lin et al. [48] demonstrated that tapping the same rhythm on both pairing devices can create a shared secret key. Any tappable input device, from a simple button to various sensors, can utilize the method. However, these methods may be vulnerable to a shoulder-surfing attack.

In general, establishing a connection between two mobile devices is termed *device pairing*. As devices have become wireless, the communication medium no longer dictates which devices will connect. Instead connections are established based on user actions that can take many different forms. A widely deployed mechanism is Bluetooth pairing, which involves selection of a target device from a list of available devices. However, researchers have demonstrated an alternative approach that also uses many different types out-of-band channels: laser-based [49], button-based [50], movement-based [51], touch-based [52] audio-based [53], light-based [54] and so on. However, a primary purpose of these OOB channels is to select a specific device in the environment, and they do not consider how to protect the established connection.

Studer et al. [55] developed a secure pairing protocol, Shot, that also used an accelerometer to authenticate two smart phones. In order to transfer the authentication data, two phones must be hold together, and one phone vibrates the data while another phone obtains it using an accelerometer. However, the protocol is originally designed for Bump exchange, which is no longer available, and so it relies on a centralized server.

2.2.2 Accelerometer-based Gesture Input

Gesture-based interaction is one of the active and attractive fields of HCI. Significant number of research have been investigating various aspects of the area of gesture interaction, including how to recognize gestures, how to perform gestures, how to select gestures and so on. In the context of this research, we have considered more on the OOB channel than gesture related issues. In other words, we have tried to evaluate whether a gesture input can be used as OOB channel or not.

However, our research is closely related to gesture input involving mobile devices with accelerometers. Because many mobile devices are too small to be equipped with conventional input peripherals, such as a keyboard, gestures hold promise as an alternative input method for such devices. Jones et al. [56] developed a tilt gesture-based text entry system for mobile devices, and achieved a mean error rate of 0.09 errors per character. Choi et al. [57] were able to recognize nine digits and five symbols that are *written in the air* as an input to a phone handset. Similar work was also carried out by Agrawal et al. [58]. When drawn in the air with the smart phone, an image is generated from the acceleration data.

Gesture inputs have also been used for security purposes. Patel et al. [59] developed a system in which a shake gesture enables to access public terminals. Chong et al. [60] proposed a system that uses accelerometer-based gestures to input a password to access the mobile device. However, the aim of our research is to transfer the authentication data from one mobile device to another using gestures.

2.3 Proposed Gesture-based OOB Channel

We have proposed a new OOB channel that uses an accelerometer-based gesture input. To transfer the authentication data between two mobile devices, a user performs a sequence of gestures, which are informed by the requester, holding the verifier that has an embedded tri-axis accelerometer. Because the gesture-based OOB channel has a low-bandwidth, SAS protocol [14], explained in section 1.1, is the most suitable. However, we suggest a modified SAS protocol [23] instead as it can improve the effectiveness of the original protocol by changing a unidirectional authentication scheme into bidirectional. In other words, with the modified SAS protocol, both devices are able to be authenticated with a single OOB channel transfer.

The modified SAS protocol is depicted in Figure 2.2. As can be easily seen, it is very similar to the original SAS protocol. The main difference is that a public key of the verifier (PK_V) also takes part in the calculation of the SAS authentication data ($SAS = PK_V \oplus H(PK_V, PK_R)$), where $H()$ is any cryptographic hash function. In addition, once the verifier decides an outcome (b) of the authentication by comparing the received SAS data with its own computed one, it has to inform the requester of the result in order to finalize the bidirectional authentication. To do that, the protocol suggests another d2h+h2d

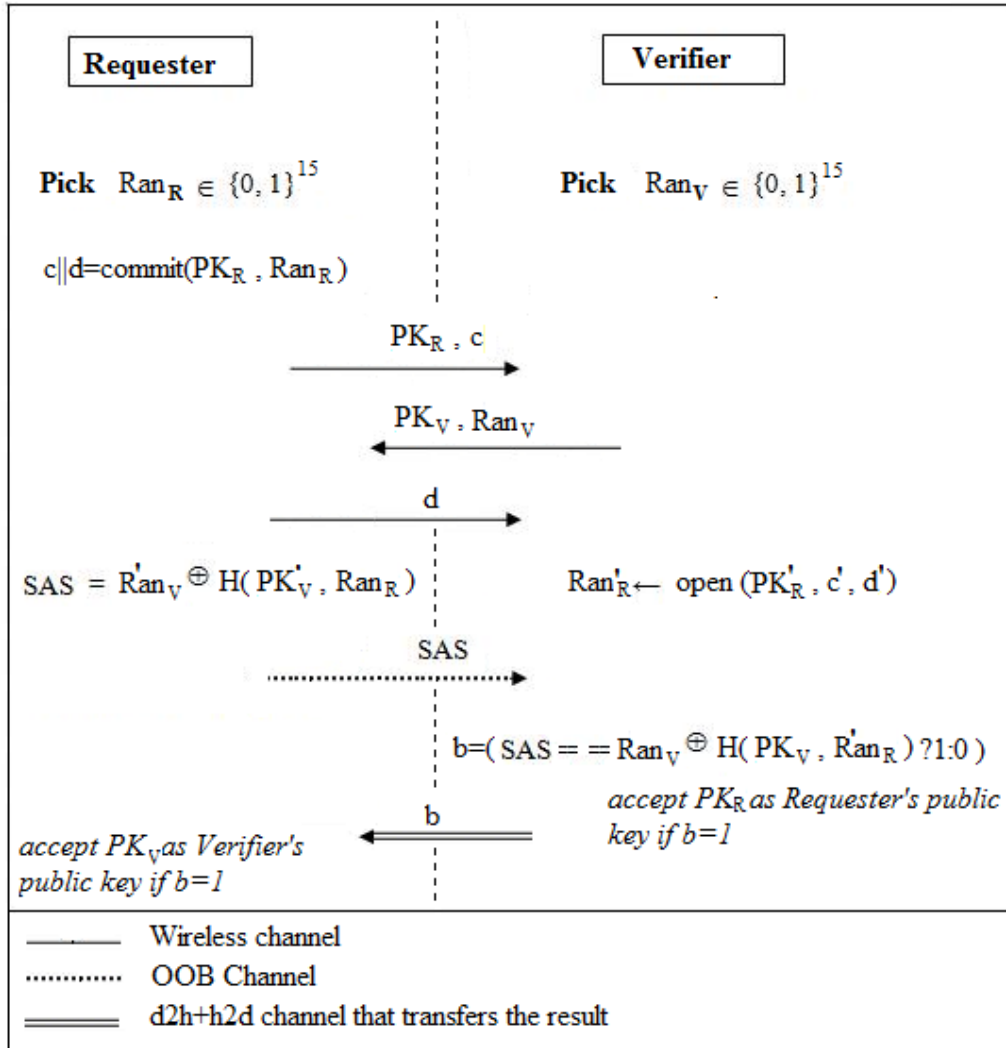


Figure 2.2: Modified SAS protocol

channel, so that the result can securely reach the requester. However, the result is either yes or no; therefore, it should not be a problem.

From the user’s perspective, the authentication process of the modified SAS protocol follows six steps, as shown in Figure 2.3. Steps 1, 5, and 6 are the same in all OOB channels whereas steps 2, 3 and 4 vary depending on the channel. Each step is described in detail, as follows.

STEP 1. Public keys and other necessary data for the SAS protocol are exchanged via a regular wireless channel between two devices.

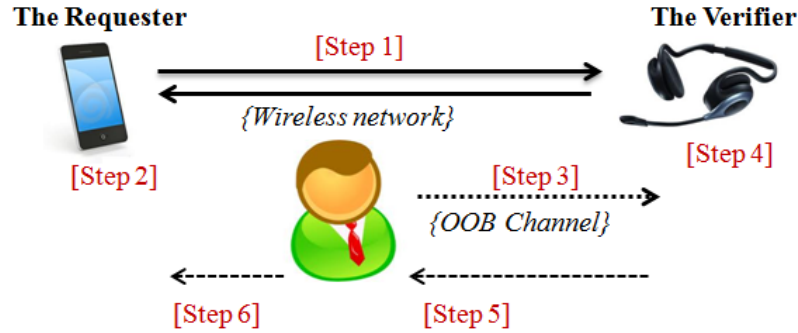


Figure 2.3: Authentication process in the gesture-based OOB channel

- STEP 2. The requester device computes the SAS data, converts them into gestures, and informs the user of the gestures. Any means can be used so long as it tells the user which gestures he or she must perform. For example, if the requester device has a graphical display, it can show the gestures on the screen. If the requester can report only numbers, an identification number (ID) can be used for the gestures. In this case, a hardcopy or web-based manual is needed for the user to look up the corresponding gesture images.
- STEP 3. The user performs the gestures with the verifier device that has an embedded tri-axis accelerometer.
- STEP 4. The verifier device recognizes the gestures, converts them back into the data, and compares the data with own created SAS data.
- STEP 5. The verifier device informs the user of the result of this comparison. The result is either YES or NO, where any simple output, such as the blink of an LED light or a beep, is sufficient.
- STEP 6. The user informs the requester device of the result. It also can be done with a simple action, such as pressing a button.

2.4 Prototype System

To assess the usability of gesture-based OOB channel, we have developed a prototype system and conducted two user experiments. In the prototype, we used a desktop PC as the requester and an iPhone 4S handset (with a dual-core 1 GHz Cortex-A9 CPU with 515

MB of RAM and an STMicro 3-axis accelerometer with a 100 Hz output data rate) as the verifier.



Figure 2.4: Prototype setup of the user experiment

The main task of the prototype system is to transfer 15 bit data from the PC to the phone handset using a gesture input. Figure 2.4 illustrates how the transmission occurs in the prototype system. First, the PC generates a random 15 bit number, converts it into a sequence of gestures, and displays both the number and the gesture set. Following this, a user carries out the gestures, one by one, holding the handset. Once the gestures are performed, the phone handset converts them into a number, and displays the number on its screen. If the original number on the PC screen and the number on the handset match, the transmission is successfully completed.











2.4.1 Gesture Library

Because the user has to perform a sequence of multiple gestures for one input, we chose gestures that end in the same place they start. Also, we tried to avoid too complicated gestures for the sake of easiness. Table 2.1 presents selected gestures, their names and corresponding IDs in the system. The start of each gesture is marked with a black dot and the end is marked with an arrow.

2.4.2 Gesture Encoding

To obtain a relationship between the size of the gesture library and the performance of the channel, four different interfaces are proposed: Digit10, Bit8, Depth6, and Bit4. The

Table 2.1: List of selected gestures

ID	Gesture	Name	ID	Gesture	Name
0		Right&Back	5		Left Circle
1		Down&Back	6		Right Square
2		Left&Back	7		Left Square
3		Up&Back	8		Right Triangle
4		Right Circle	9		Left Triangle

gesture library of the Bit4 interface consists of the first four gestures in Table 2.1, while Depth6, Bit8, and Digit10 consist of six, eight, and ten gestures, respectively. Table 2.2 summarizes four interfaces.

Table 2.2: Summary of proposed interfaces

Interface	Gesture Library	Gestures per Input
<i>Bit4</i>	0, 1, 2, 3	8
<i>Depth6</i>	Bit4 + 4, 5	6-10
<i>Bit8</i>	Depth6 + 6, 7	5
<i>Digit10</i>	Bit8 + 8, 9	5

To convert 15 bit data into a sequence of gestures, each interface utilizes a different encoding method based on their library sizes. These conversions were implemented using .NET framework and run on the desktop PC.


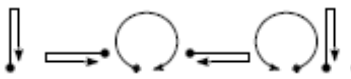
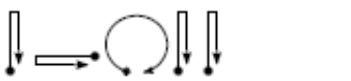

Table 2.3 provides an example of how each interface converts decimal number (13595) into gestures. Conversion methods are explained as follows.

Digit10: It changes each digit, or ID in our case, with a corresponding gesture directly.

Bit8: The number is converted into an octal number and each octal digit is changed with a corresponding gesture.

Depth6: The number is also converted into an octal number. Eight octal digits are divided into two levels, as shown in Figure 2.5, where four gestures represent

Table 2.3: Gesture conversion sample of interfaces

Interface	Gesture Conversion (13595 ₁₀)
<i>Bit4</i>	03110123 ₄ 
<i>Depth6</i>	32433 ₈ 
<i>Bit8</i>	32433 ₈ 
<i>Digit10</i>	13595 ₁₀ 

digits in each level and the remaining two gestures are used to move between levels.

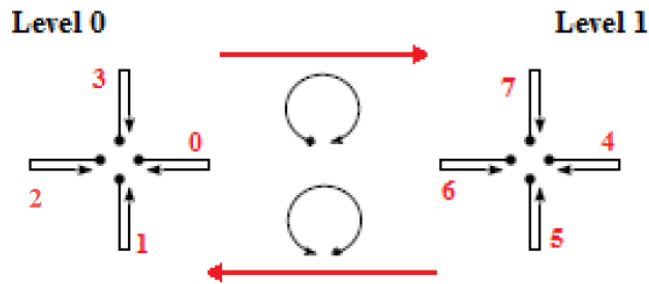


Figure 2.5: Depth6 conversion method

Bit4: It applies the same encoding as Bit8, except the number is converted into a quaternary number.

To obtain the data back from gestures, the very same methods, but in reverse order, are applied in each interface. These conversions were implemented in Objective C language and run on the iPhone handset.

2.4.3 Gesture Recognition

The first step of accelerometer-based gesture recognition is to get the acceleration data of a gesture motion, and then the gesture is recognized from the captured data. Researchers have applied diverse techniques, including the hidden Markov method (HMM) [61–63], Bayesian networks [64], FDSVM [65], and dynamic time wrapping (DTW) [66–68] to the

accelerometer-based gesture recognition. We used the DTW method in our prototype because of its comparatively high accuracy, fast computational performance, and the need for relatively few templates [65, 69].

DTW is an algorithm that measures the similarity between two sequences of different lengths [70]. It has long been known in speech recognition community [71], however, the technique has flourished, and has been applied to a variety of problems, including gesture recognition. Because DTW is a template-based method, gesture recognition selects the best-matched template as a candidate. To determine the candidate template for a gesture from a predefined template library, the similarity costs between the gesture and each template are computed first. Following this, the template that has the lowest similarity cost is chosen as the candidate.

In our prototype, the acceleration data is captured in 10ms (or 100Hz) intervals. Prior to computing the similarity cost, a couple of optimization methods are applied to the raw data. First, the acceleration data is filtered using a high-pass filter to isolate sudden changes in movement from the constant effects of gravity. We then quantize the filtered data by an averaging window of 50 mps that moves at a 30 mps step. Quantization reduces the length of the data as well as improves recognition accuracy by removing acceleration noise and minor hand tilt. In addition to the optimization methods, we use an exact half of the data for DTW to further improve the computational efficiency. Because our selected gestures are symmetric, no change in the gesture recognition is observed between using the full data and the half data.

2.4.4 Dynamic Time Wrapping Algorithm

Here, the DTW algorithm is explained briefly. Assume that the two sequence, \mathbf{p} and \mathbf{q} , are of length n and m respectively, where $\mathbf{p} = p_1, \dots, p_n$ and $\mathbf{q} = q_1, \dots, q_m$. To compute the similarity cost of the two sequence, DTW (\mathbf{p}, \mathbf{q}), we first construct an $n \times m$ matrix, where the (i,j) -th element of the matrix indicates the distance $d(p_i, q_j)$ between two points p_i and q_j , where $d(p_i, q_j) = (p_i - q_j)^2$. The cost of similarity between the two sequence then is found by applying the dynamic programming formulation below, which defines the cumulative cost $D_{i,j}$ as the cost $d(p_i, q_j)$ in the current cell plus the minimum of the cumulative cost of the adjacent elements,

$$D_{i,j} = d(p_i, q_j) + \min(D_{i,j-1}, D_{i-1,j}, D_{i-1,j-1}) \quad (2.1)$$

and consequently,

$$DTW(\mathbf{p}, \mathbf{q}) = D_{n,m} \quad (2.2)$$

An example of how we computed the similarity cost between two gestures using DTW is given here. In our case, the tri-axis accelerometer measures acceleration in three directions: x, y, and z. Each point in a sequence, \mathbf{p} , then consists of three independent values, $p_i = (p_{i,x}, p_{i,y}, p_{i,z})$. The distance $d(p_i, q_j)$ between two points p_i and q_j of two gesture sequences \mathbf{p} , \mathbf{q} then becomes

$$d(p_i, q_j) = (p_{i,x} - q_{j,x})^2 + (p_{i,y} - q_{j,y})^2 + (p_{i,z} - q_{j,z})^2 \quad (2.3)$$

Let \mathbf{p} as a predefined template of ‘Right&Back’ gesture in our experiment, and let \mathbf{q} as a result of a participant performing the same gesture. The table below presents the optimized as well as cut in half sequences of two gestures.

	\mathbf{p}	\mathbf{q}
1st	(-0.29, -0.20, 15.99)	(-0.57, -0.17, 16.05)
2nd	(-0.44, -0.01, 14.64)	(-0.29, -0.19, 14.26)
3rd	(-0.60, -0.12, 15.69)	(-0.26, -0.30, 15.43)
...
34th	(0.67, -0.16, 15.03)	...
35th	(0.53, -0.18, 14.97)	...
...
43th		(-0.13, -0.51, 15.11)
44th		(-0.09, -0.48, 15.01)

In this case, $n=35$ and $m=44$, and we start by constructing a matrix that is 35×44 and placing \mathbf{p} and \mathbf{q} on each side of the matrix, as shown in Table 2.4. The (i, j) -th element of the matrix is the distance of the two points, calculated by formulation 2.3.

Once the first matrix is filled, a second matrix is constructed. The (i, j) -th element of the second matrix represents the cumulative cost of the current cell and it is computed by applying the formulation 2.1, as shown in Table 2.5. The similarity cost between the two sequences \mathbf{p} and \mathbf{q} , $DTW(\mathbf{p}, \mathbf{q})$, is equal to the cost in the top right corner of the matrix that is 10.489 in this example.

Table 2.4: Constructing the distance matrix from two sequences

p points						
35th	2.362	1.179	0.842	...	0.563	0.473
34th	2.578	1.513	1.045	...	0.774	0.683
...
3rd	0.130	2.138	0.215	...	0.710	0.851
2nd	2.035	0.202	0.745	...	0.590	0.495
1st	0.086	2.986	0.330	...	0.895	1.080
	1st	2nd	3rd	...	43th	44th
						q points

Table 2.5: Estimating the similarity cost of two sequences

78.867	46.341	33.151	...	10.016	10.489
76.505	45.163	32.309	...	11.557	12.240
...
2.251	2.426	0.503	...	32.675	33.407
2.121	0.288	1.032	...	32.556	33.050
0.086	3.073	3.403	...	59.983	61.062

2.4.5 User Interface

In our prototype, the iPhone handset performs the gesture recognition, and so it was implemented using Objective C language. To store the acceleration data of the templates, we used SQLite database library [72], which is free as well as suitable for memory constrained mobile devices. Because we have conducted two different experiments, two slightly different systems were developed.

Figure 2.6 shows the user interface of the first experiment. The goal of the experiment is to figure out which library size is the most efficient for the gesture-based OOB channel; therefore, the user interface consists of 4 parts: Bit4, Depth6, Bit8 and Digit10. Each interface is basically the same, except the number of gestures to perform. Figure 2.6 (a) illustrates how the desktop PC informs the participants of gestures to carry out. It also displays the original number and an execution order of four interfaces. For each generated number, the participants can perform the gesture input of all four interfaces. Therefore, the execution order of the interfaces is also randomly created for each time to avoid the interface bias.

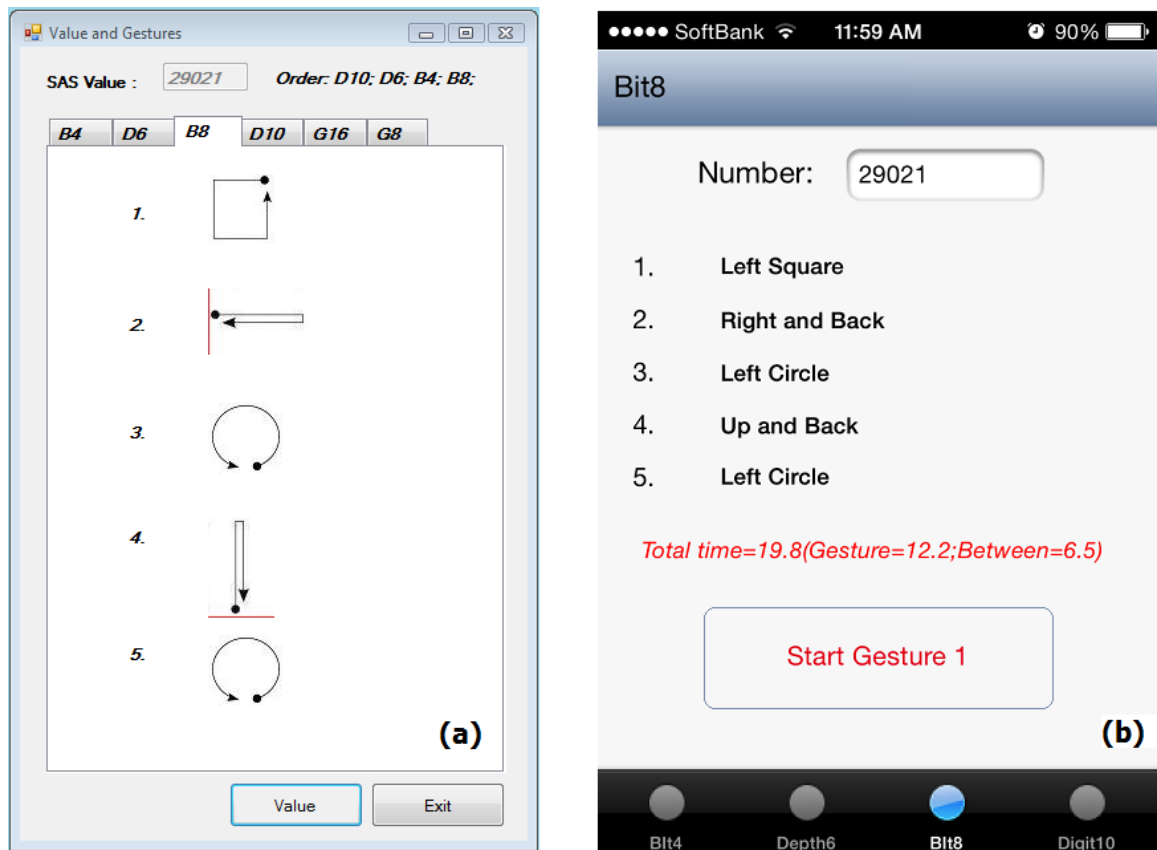


Figure 2.6: User interface of experiment one: (a)the requester side - desktop PC (b)the verifier side - iPhone handset

Figure 2.6 (b) illustrates the interface of the iPhone handset. It has a button, so that the participants can press it before starting and after ending the gesture to distinguish it from other unwanted movements. An alternative is that the users simply shake, tilt, or tap their mobile device instead of pressing the button. Once the gestures are performed, it displays the names of each recognized gestures, the converted number, and the total time it has spent.

Figure 2.7 presents the user interface of the second experiment. The aim of the experiment is to determine if user-defined templates improve the accuracy of the gesture-based OOB channel. The user interface consists of two parts: evaluation and user list. The 'Evaluation' part is the main interface for executing the task, and so it is designed in such a way that the participants can select the user-defined templates from the list (Figure 2.7 (a)). The 'User List' part allows the participants to input their own gesture templates into the

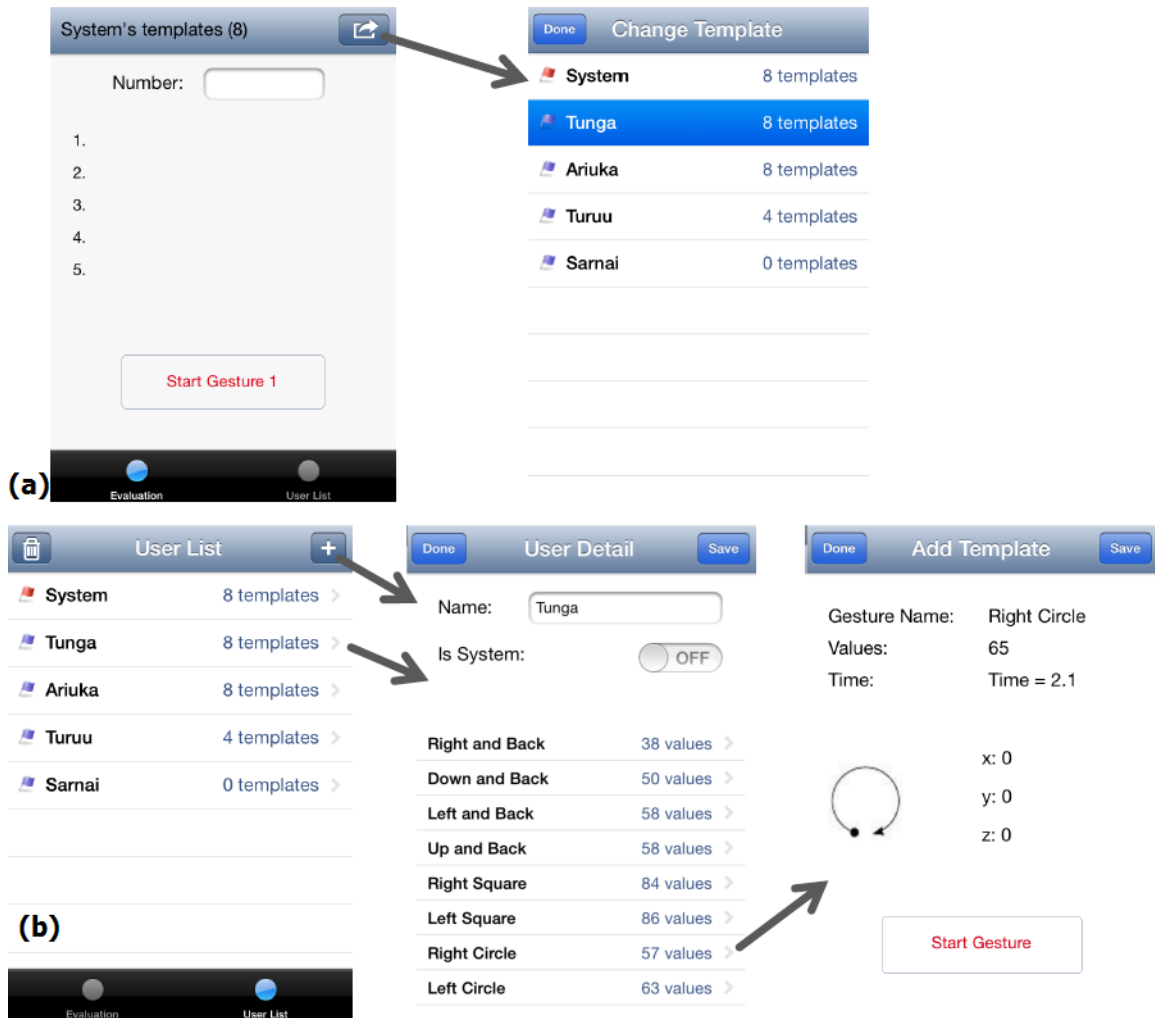


Figure 2.7: User interface of experiment two: (a) evaluation (b) user list

system. To do that, the participants carry out each gesture template the very same way as they do the gesture input, and this time, the image of the gesture is displayed on the screen of the handset (Figure 2.7 (b)).

2.5 Experiment One: Usability Analysis

2.5.1 Purpose

In this experiment, we investigated the usability of the gesture-based OOB channel, specifically the completion time, error rate, and learnability. At the same time, we also

want to determine the most appropriate size of library for gesture-based OOB channel. Two hypotheses were formulated. First, there is no difference between the interfaces in terms of completion time and error rates. Second, there is no difference between the rounds in terms of the completion time and error rates.

2.5.2 Participants and Task

Ten volunteers (three females and seven males) participated in our experiment; all of them were either undergraduate or graduate students, and all were right-handed. Their mean age was 24.2 years (SD of 1.9 years, range 22-29)

Before beginning the experiment, a brief introduction was given on how to carry on a gesture input with the phone and what kind of gestures were to be used. We suggested the participants hold the phone in the same way as shown in Figure 2.8, so that their thumb can easily press the button on its screen as well as they can easily perform gestures in the two-dimensional (vertical and horizontal) space.



Figure 2.8: Phone holding in the experiment

The participants were allowed to practice one time. Every participant performed the gesture inputs 12 times with three rounds for each interface. In each round, the order of the interfaces was randomly generated and queued to reduce bias for certain interfaces. The experiment lasted for approximately 20 minutes. The total input time was measured from the beginning of the first gesture to the completion of the final gesture.

With the 10 participants, a total of 120 gesture inputs were entered during the experiment (four interfaces and three rounds). To analyze the collected data, repeated measures

analysis of variance (ANOVA) was used. If the results were statistically significant, we further analyzed the data using a paired t-test between every two interfaces to locate the differences.

2.5.3 Completion Time

Table 2.6 summarizes the completion time of the our gesture-based OOB channel transfer for each designed interfaces. Repeated measures ANOVA with interface type as the factor was significant with $F(3, 29) = 45.95$ and $p = 0.0000$. The correlation between the completion time and the size of the gesture library was a negative high, $r = -0.812$.

Table 2.6: Completion time of transmission (sec)

Interface	Mean	SD	Min	Max
Bit4	22.90	5.69	11.80	36.20
Depth6	25.02	7.11	11.60	38.80
Bit8	16.86	4.23	8.30	25.30
Digit10	16.76	4.79	8.10	28.20

Therefore, we conducted paired t-tests between each pair of interfaces, as presented in Table 2.7. The results suggested that the transmission speed between interfaces differed significantly ($t < 2$, $p < 0.05$) for most pairs, except Bit8 and Digit10 combinations ($t = 0.012$, $p = 0.904$).

Table 2.7: Paired tTest Results of the completion time

	Bit4	Depth6	Bit8	Digit10
Bit4	—	$t = -2.28$ $p = 0.029$	$t = 8.47$ $p = 0.000$	$t = 7.55$ $p = 0.000$
Depth6	$t = -2.28$ $p = 0.029$	—	$t = 7.79$ $p = 0.000$	$t = 9.29$ $p = 0.000$
Bit8	$t = 8.47$ $p = 0.000$	$t = 7.79$ $p = 0.000$	—	$t = 0.012$ $p = 0.905$
Digit10	$t = 7.55$ $p = 0.000$	$t = 9.29$ $p = 0.000$	$t = 0.012$ $p = 0.905$	—

Figure 2.9 shows the distribution of the mean times for performing the gestures, gaps between the gestures, and the gesture recognition time of each interface. The Bit4 and

Depth6 interfaces required more time for user input but less time for gesture recognition compared to the Bit8 and Digit10 interfaces.

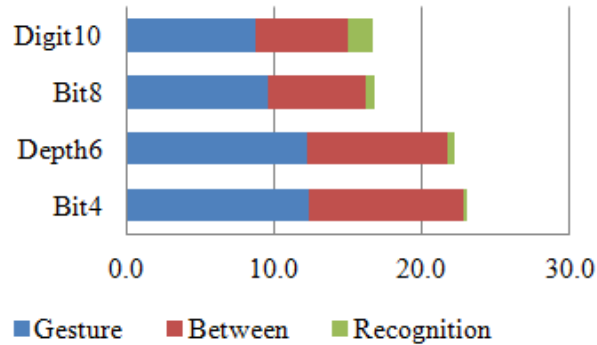


Figure 2.9: Time distributions for a gesture input

As a result of this evaluation, we concluded that the completion time decreases as the size of gesture library increase. In other words, the more gestures the system has, the less time it spends on transmission. However, both 8 and 10 gesture libraries can be the candidate for the size of gesture-based OOB channel because they required almost same amount of time.

2.5.4 Error rate

The error rate of the four interfaces is summarized in Figure 2.10. The Digit10 interface exhibited the greatest error rate, with a mean error per input of 0.33 and $SD = 0.48$, with 0.06 errors per gesture, whereas Bit4 had the fewest, with a mean error per gesture of 0.06 and $SD = 0.25$, with 0.008 errors per gesture. The mean error rates of Depth6 and Bit8 were similar, at 0.1 and 0.13 errors per input and 0.014 and 0.019 per gesture, respectively.

Repeated measures ANOVA tests with interface as a factor were significant with $F(3, 29) = 4.43$ and $p = 0.006$. We conducted paired t-tests between the error rates of the different interfaces. There was no significant difference between Bit4, Depth8, and Bit8 ($p > 0.5$). However, Digit10 differed significantly from the other interfaces, with $t < 2$ and $p < 0.05$. The correlation between the error rate and the size of the gesture library was quite high with $r = 0.897$.

As a result of this evaluation, we concluded that the probability of successful transmission increases as the size of gesture library decreases. In other words, the more gestures the

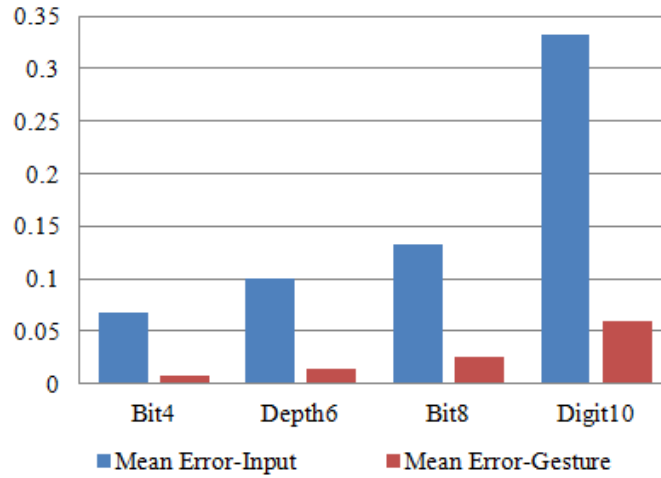


Figure 2.10: Mean error rates

system has, the more error-prone it becomes. However, all of 4, 6, 8 gesture libraries can be the candidate for the size of gesture-based OOB channel because they showed very close error rates.

2.5.5 Learnability

The participants performed three rounds of tasks. Table 2.8 summarizes the speed of gesture input for each round. Repeated measures ANOVA tests with round as the factor showed that there was a significant difference between the rounds with $F(2, 38) = 8.48$ and $p = 0.004$. However, there were no significant differences in terms of error rates between the rounds.

Table 2.8: Completion time of rounds(sec)

Round	Mean	SD	Min	Max
First	21.7	11.7	10.3	39.8
Second	20.9	6.7	8.1	38.4
Third	18.8	7.2	8.3	32.5

As a result of this evaluation, we concluded that a user may learn how to perform the authentication using gesture-based OOB channel over a short period of time. However, the success of the transmission does not depend on the user.

2.5.6 Subjective Ratings

Each participant was asked to complete a brief questionnaire. Following the experiment, they were not able to clearly distinguish one interface from another. Therefore, the questions were on the overall impressions of the gesture input. The questionnaire consists of the following three questions.

Q1: How do you rate the ease of performing the gesture input?

Q2: If this gesture input enable secure communication between your mobile devices, how do you rate the amount of time spend on one input?

Q3: If you have to do perform the gesture input next time, could you carry out it effectively?

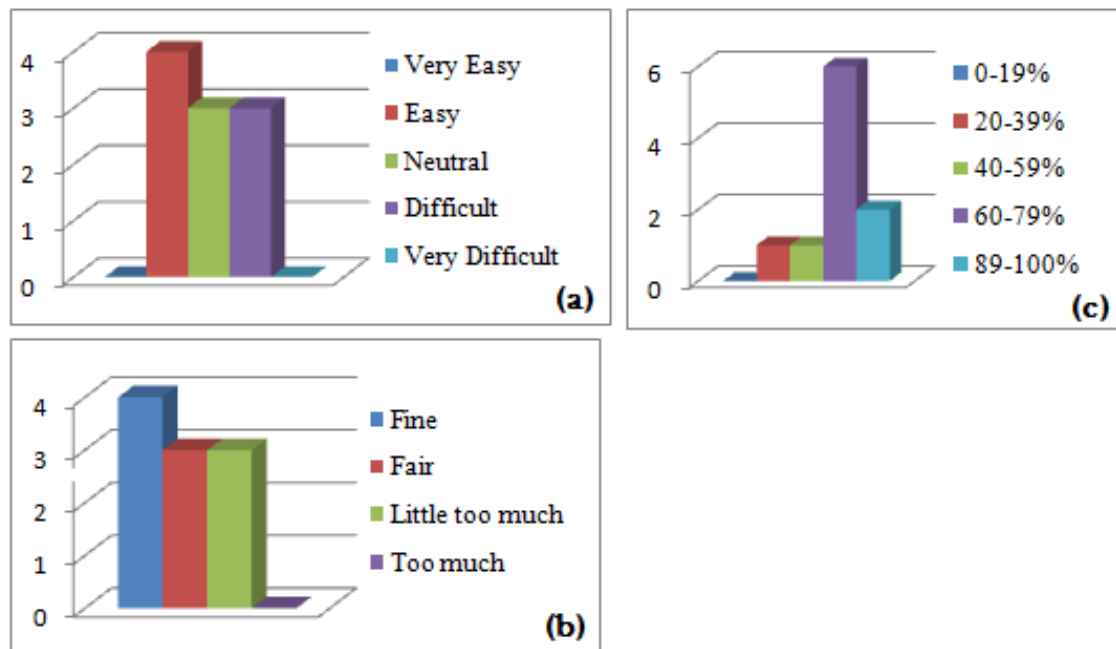


Figure 2.11: Result of users' impression on the gesture input: (a) easiness (b) satisfactory of time (c) sureness of next successful try

In first question, participants asked to give their rating on ease of use with scale of five intervals. Figure 2.11 (a) summarizes the result, and the seventy percentages of the participants had a positive attitude toward ease of the gesture input. In second question, we asked about their impression on completion time of the gesture input with scale of four intervals. The seventy percentages of the participants were also satisfied with the time they

spent (Figure 2.11 (b)). In third question, we asked their opinion about if they can perform gesture input effectively next time with scale of five intervals because each participant has performed 12 gesture inputs and had enough idea of how to do it. More than eighty percentages of participants were quite sure of themselves to carry out gesture input (Figure 2.11 (c)).

2.6 Experiment Two: User-defined Templates

2.6.1 Purpose

In the second experiment, we investigated whether or not the error rate decreased if users inserted their own gesture templates into the mobile device. It is quite possible that our gesture-based OOB can be extended so that users can carry out the required gestures with the verifier device before they start the authentication. In the previous experiment, the Bit8 interface showed the best results, so we selected Bit8 as the base interface for the second experiment. Only two interfaces were implemented: one used predefined templates while the other used user-defined templates when identifying the gestures.

2.6.2 Result

A different set of 10 volunteers participated in the second experiment (two males and eight females, all of whom were right-handed). Prior to commencing the experiment, each participant inserted the first eight gestures shown in Table 1 into the phone by performing them to create their own templates. Once all eight templates were completed, the participant carried out the gesture input six times (three rounds for each interface).

We assessed whether there would be a difference between using the predefined templates and the user-defined templates in terms of the error rate. From the 10 participants, we collected a total of 60 gesture inputs (three rounds and two interfaces). Because we only had two groups of data to compare, we used a paired t-test instead of repeated measures ANOVA. However, no significant difference was found between the two interfaces ($t(29) = 1.79$, $p = 0.083$).

2.7 Discussion

As shown in the table below, it appears that the Bit8 interface is the best choice for our gesture-based OOB channel, considering the relatively high negative correlation between the completion time and the size of the gesture library, and positive correlation between the error rate and the size of the gesture library, as well as the similar completion times of Bit8 and Digit10 interfaces, and error rates of Bit4, Depth6, and Bit8 interfaces.

	Bit4	Depth6	Bit8	Digit10
<i>Completion Time</i>	×	×	○	○
<i>Error Rate</i>	○	○	○	×

The transmission speed of Bit8 interface is 16,86 seconds that is faster than some of the popular OOB channels, such as Barcode (37 s), Alphanumeric (40 s), BEDA (45 s), Beep&Blink combinations (30 s), and Loud&Clear (20 s). Furthermore, the Bit8 interface has a mean error rate of 13%, which is lower than that of Beep&Blink combinations (about 2030%), Barcode (53%), Alphanumeric Copy&Enter (23%), many versions of Compare&Confirm (1636%), Alphanumeric Compare&Select (30%), and Numeric Copy&Enter (13%) [36, 44].

Library size of 8 gestures is the most effective among our tested sizes. This result may be true in general because as the library size of gesture-based OOB channel increases, completion time decreases and error rate increases. However, the effectiveness of our channel, especially error rate, could be different in some situations since it depends profoundly on gesture related issues, such as gesture recognition techniques, gesture selection, gesture type and etc. Good news is that our gesture-based OOB channel can be improved by using advances in gesture related research.

In our experiments, there was no difference between Digit10 and Bit8, and only a small difference between Bit4 and Depth6 in terms of completion times. In other words, the number of gestures per input had greater influence on the completion time than did the size of the gesture library, because gesture recognition took less time than the users actions.

The error increased dramatically when using Depth10 (as shown in Fig. 5). The size of the gesture library may be one factor ($r = 0.897$). Our log file revealed that approximately 80% of the Depth10 errors were related to triangle gestures. It follows that the selection of proper gestures is very important.

As shown in Table 2.8, the mean completion time decreased over trials in a short period of time while the accuracy remained the same. Moreover, participants became more confident in the execution of these tasks over the course of the experiment.

One unexpected finding is that the user-defined templates did not improve the accuracy of the channel. However, these results should be interpreted with caution as the sample size was relatively small. A similar study [73], for instance, has found that users interact more accurately with a media player when adjusting predefined gestures. However, our participants asked to adjust templates of the same gestures with their own whereas UbiGesture system [73] considered the case that the users change system gestures with the completely different gestures.

Almost two-thirds of the participants had a positive attitude with respect to the ease of the method, and 70% were satisfied with the time required.

During the experiments, we made a number of important observations that may help to improve our gesture-based OOB channel. First, the speed of the users varied, which noticeably affected the success of gesture input. Therefore, the gesture-based OOB channel should be able to adapt to user speed by providing feedback. Secondly, the sequence of gestures was carried out in a discrete manner; however, participants preferred continuous gestures. In other words, pressing a button before and after every gesture placed an additional burden on the user. Finally, the participants indicated that they did not like holding the phone in a fixed position.

2.8 Summary

This chapter introduced our proposed OOB channel that uses an accelerometer-based gesture input. Gesture-based OOB channel is suitable for all-kinds of mobile devices, including input/output constrained devices, as the accelerometer is small and incurs only a small computational overhead. In our OOB channel, one device converts the authentication data into a sequence of gestures and informs a user of them. The user then performs the gestures one-by-one with the second device that has an embedded tri-axis accelerometer to transfer the data into it.

We have implemented a prototype system and conducted thorough usability analysis to compare our gesture-based OOB channel with the existing OOB channels. Result showed that the gesture-base OOB channel has reliable performance with a mean completion time

of 16.86 seconds with $SD=4.2$ seconds, as well as a mean error rate of 0.13 per transfer with $SD = 0.34$. Four different gesture sets were used in this experiment to determine which library size would be more usable in practice. The result suggested that library size of 8 gestures was the most effective among our tested sizes. We have also conducted another experiment to determine whether user-defined gesture templates improve the accuracy of gesture recognition. Contrary to expectations, this study did not find a significant difference between the predefined and user-defined templates in terms of the error rate.

Chapter 3

Accumulative Secure Group Association

3.1 Motivation

To support our research vision, “the more OOB channels exist, the more security is available”, we have designed a new Secure Group Association (SGA) method, in which mobile devices could employ their desired OOB channels. In contrast, the existing group security protocols require all group members to use the same or similar OOB channels. We imply that our proposed method that encourages various OOB channels has more chance of successful association than the existing SGA protocols that are too strict for spontaneous cases. Furthermore, unlike the existing SGA protocols, the proposed method associates devices in an accumulative manner, which means that it does not require all group members to be present in one place at one time.

There are a couple of important reasons why we have considered the secure group association, especially the accumulative method.

First, little attention has been paid to the association among a group of mobile devices, even though there is a considerable body of literature in the field of secure pairing of two mobile devices. However, there are numerous occasions in the modern wireless world in where a group of mobile devices need to interact spontaneously as well as securely with one another, without any prior preparation. Members in a meeting, for example, establish a group network to exchange sensitive resources. In addition, more and more collaborative

tasks are shifting or are likely to shift toward mobile devices. For instance, consider a scenario that two groups of friends decide to play a game and will compete against each other. Members of each team want to set up secure group, so that they can privately discuss strategy without the other group overhearing them.

Second, few group security protocols have been proposed; however, these have tended to focus on a scenario whereby all devices must be located in one place to establish the secure connections. These simultaneous associations have several drawbacks, such as a lack of scalability, limited group size, poor device diversity and etc. Whereas, our accumulative secure group association method is designed for a broader range of scenario and addressed all these limitations.

3.2 Related Work

3.2.1 Secure Group Association Protocols

To date, there have been relatively few studies in the field of secure group association; however, a number of group security protocols including Multi-Party [74], SAS-GMA [75], HCBK [76], and SPATE [77] have been proposed. In simple terms, each device sends its public key to every other device via wireless networks, and then each device generates the authentication data independently. Authentication is successful if all of the generated data are the same. However, this approach has some significant limitations. First, all mobile devices must have the proper output for data to be compared. Second, verifying the data from all devices is tedious, and becomes even more so as the number of devices increases.

Chen et al. [78] introduced a slightly different SGA protocol, called GAnGS. Because previous methods are difficult to implement with large groups, they divides groups into subgroups for verification. However, the protocol uses a barcode-based OOB channel, which requires all devices to have both a camera and a display. Moreover, subgrouping increases the burden on users.

When talking about the user-aided group authentication, two important aspects must be considered: what kind of OOB channel the method utilizes and how much the user involvement the protocol requires. These two aspects of the existing SGA protocols are presented in Table 3.1. If a single member broadcasts its authentication data and the remaining members compare it with their own, the number of OOB channel transfers can

be $(n-1)$, where n is the size of the group, for SAS-GMA and Multi-Party. Furthermore, in order to detect intruders, group security protocols basically require the users to verify the size of the group by counting the potential group members, and then either by inputting it to the system before the association or by comparing it with the number of group members after the association. The size verification is not included in SAS-GMA and HCBK, and that make these protocols more impotent than others. Moreover, GAnGS uses a barcode-based OOB channel to collect group members' information in one place.

Table 3.1: Summary of the existing SGA protocols

Protocols	Channel Type	User Involvement	
		OOB Transfer	Other
Multy-Party	Screen	$\geq (n - 1)$	Size verification
SAS-GMA	Not specified	$\geq (n - 1)$	-
SPATE	Screen	$n-1$	Size verification
HCBK	Screen	$n-1$	-
GAnGS	Screen & Camera	$n-1$	Size verification Data collection

Recently, few attempts [79, 80] have been made to apply a group protocol SAS-GMA [75] to the current OOB channels and to investigate the practical usability of group association methods. However, these studies used three and four OOB channels respectively, which require rich user interfaces, as most OOB channels are not suitable for the protocol. In addition, one of the significant findings from these studies is that many failures are caused by miscommunication between users, even in a small group (of fewer than six).

Several group security protocols for body area network (BAN) in where a group consists of many low-capability sensor devices and one controller device that has richer user interfaces and higher computational power, and work together toward monitoring patients health in healthcare systems, has been proposed [81–83]. BAN may be considered as a specific type of mobile network that needs high level of privacy and security in its interactions. However, it lacks ad-hoc nature of mobile groups in general, and so it was reflected in designed methods. Keoh et al. [82] have proposed a system ,for example, that requires a centralized trusted third party and assumes that every device and employees are certified by the hospital before deployment.

3.2.2 Group Association

A number of attempts have been made to associate multiple mobile devices, which considers about selecting the potential group members from the environment. Lucero et al. [84] have developed a method whereby during group association, each device must touch the device to the right of it. This method is easy, fast, and also can prevent an adversary from joining. Chong et al. [85] have proposed another simple and rapid group association technique, *GroupTap*. It utilizes NFC tag as an OOB channel, and the users tap their mobile devices on a selected object to get associated. However, an issue of how to protect the communications once the group has been associated was not addressed in these methods.

Controllability indicates a social factor; an association could be controlled by either a single user or multiple users. Uzun et al. [86] were apparently the first to introduce the concept and to use the term *social pairing* that involves two different users establishing pairing between their respective devices. They have conducted usability analysis on the existing pairing techniques that enable social-pairing. Their study suggested that people are reluctant to share their personal devices with others, especially strangers, as it raises privacy concerns. In the context of group association, the issue related to controllability becomes more crucial, because the group interactions are inherently a social activity. However, no research has been found that considered the controllability in group cases.

Other social factors of group association have been examined in recent studies. Finding in Kuo et al.'s study [87] demonstrated that group association protocol designs are situation dependent, and no single solution is appropriate for all situations. In addition, Chong et al. [88] revealed that people's choice of group association techniques are largely influenced by devices' different attributes, such as their mobility and flexibility, as well as their prior knowledge of interaction with technology.

Moreover, other techniques that form a secure group in ad-hoc and ubiquitous environment exist. For example, the concept that every node on a network issues certificates to other nodes without a trusted third party is not a new idea. It is termed a user-centric trust model, and some wired network systems that also have an ad-hoc nature, such as peer-to-peer or multi-agent systems, use this model. Therefore, it can easily be adapted to wireless networks, and the authentication essentially relies on the trust calculation [89]. However, quantifying authentication is a controversial topic in itself [90].

3.3 Secure Group Association Types

In general, SGA can be classified into all-at-once and one-by-one association. Table 3.2 compares some important characteristics of the two categories.

Table 3.2: Characteristics of all-at once and one-by-one associations

Characteristics	All-at-once	One-by-one
Literature reports	A few	None
Proximity	Required	Not required
Synchronicity	Required	Not required
Device Diversity	Limited	Partly limited
Scalability	Weak	Strong
Group Size	Limited	Unlimited
User Involvement	Full	Semi-automatic
Robustness	Weak	Strong

The characteristics of SGA types are expanded on as follows:

- *Literature reports*: There has been little discussion of SGA, and all published studies have described all-at-once group association [74–78]. The comparison in Table 3.2 is between these association protocols and our proposed method.
- *Proximity and Synchronicity*: All group members have to be physically located in the same place and must participate actively and simultaneously in all-at-once group association. In contrast, as its name implies, one-by-one association occurs in an accumulative manner, allowing each device to join the group independently of the other group members and their associations.
- *Device Diversity*: In the existing protocols, all group members are expected to possess the same physical method for verifying authentication data as shown in Table 3.1. In SAS-GMA protocol [75], even though a specific output was not mentioned, the protocol needed all devices to be equipped with similar interfaces so that users could compare authentication data. However, the verification phase of each device in our accumulative method involves only two devices: the new device and an existing group member. Therefore, only one device from the group is required to be compatible with the new device.
- *Scalability*: Scalability is a major disadvantage of all-at-one association. To add a new device to the group, all group members that are already associated must

be assembled and perform the association together. Group association should allow for the addition of a new device without all of the existing devices being physically present, and our method addresses this problem. In our method, only the new device pairs with one existing group member, and the remaining process is automatic.

- *Group Size*: As a group grows in number, all-at-once association becomes difficult to carry out and user effort increases considerably. Recent usability studies have shown that many failures can be caused by miscommunication of users, even in small groups of fewer than six users [79, 80]. In contrast, the size of a group in one-by-one association can grow as big as possible without most of the group members even being aware of the growth.
- *User Involvement*: Even though it is impossible to eliminate user involvement completely, group association should be as automatic as possible. In addition to transferring the authentication data through OOB channel, all-at-once group association also involves the users to correctly verify the number of members (Table 3.1). Moreover, there will be noticeable difference in terms of the user effort over a long period of time. The total effort for the user in all-at-once associations is increased if group reforms many times while our association remains constant.
- *Robustness*: OOB channel authentication has the relatively high possibility of encountering errors due to the user involvement [36, 44]. If an abnormal occurs during an all-at-once association, entire process aborts even it is relevant to one member. In other hand, authentication in our proposed method involves two devices only and its failure does not affect other group members and their associations.

3.4 Proposed Accumulative Secure Group Association Method

3.4.1 Base Concept

We assume that all mobile devices are equipped with a wireless network protocol, the installation of at least one OOB channel, and computational hardware that is sufficient for the basic cryptographic operations.

Assume that Bob has several mobile devices as shown in Figure 3.1. To provide secure wireless communication between his smart phone and laptop, he uses a barcode-based OOB channel [19]. In addition, he establishes a secure connection between his smart phone and his wireless headset using a gesture-based OOB channel [91]. He now wants to secure the communication channel between the laptop and the wireless headset. The following question arises: must he perform OOB pairing, which consumes a considerable amount of time and effort [36, 44] again? We believe that our method is the most appropriate to facilitate this kind of scenario.

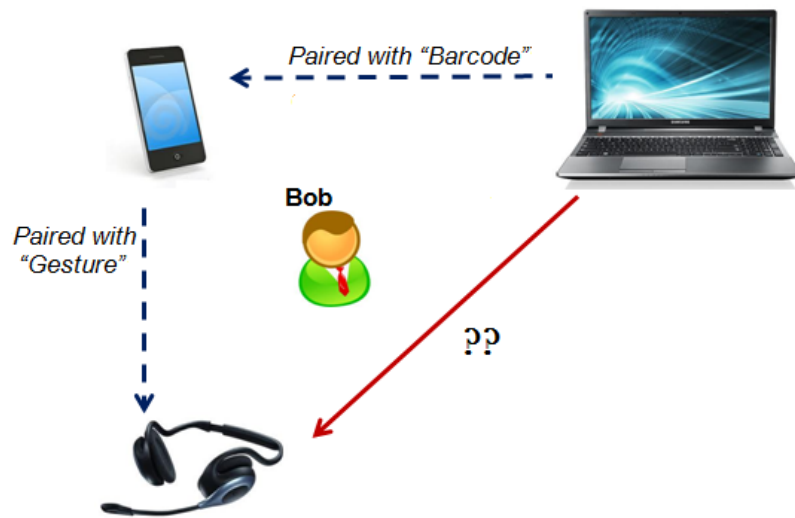


Figure 3.1: Common multiple pairing scenario

Our accumulative SGA method has two principle aspects, which are described below:

- *Certification*: Two devices are paired using user-aided authentication. Once the public key is verified successfully, the verifier issues a digital certificate to the requester, as depicted in Figure 3.2. The certificate is saved in a protected repository, called the KeyStore, on both devices. KeyStore holds two types of

certificates: certificate entries and key entries. Certificate entries are certificates that the device itself has issued to other devices, whereas key entries are used to ensure the public key of the device and are signed by other devices.

- *Certification Path Pairing (CPP)*: Saved certificates are used for subsequent pairing. For instance, suppose Bobs smart phone has already received a certificate from the laptop and that the headset has also received a certificate from the phone. This means that there is a certification path between the laptop and the headset, i.e., $Cert(Laptop \Rightarrow Phone) + Cert(Phone \Rightarrow Headset)$, as shown in Figure 3.2. Given this, the laptop can have confidence in the public key of the headset, and so issues a certificate to it without OOB channel-based pairing. Finally, a certificate is also issued to the headset from the laptop, $Cert(Laptop \Rightarrow Headset)$. If cross certification is required, CPP is repeated in the reverse direction.

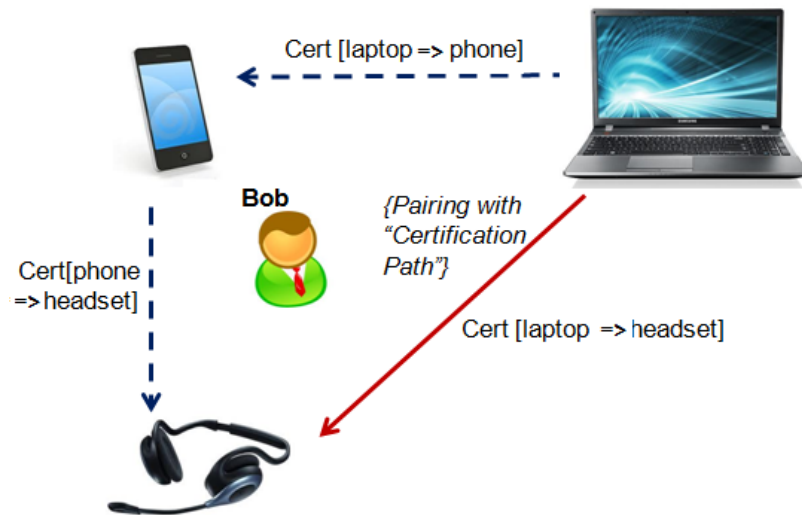


Figure 3.2: The base concept of the accumulative SGA method

Compared to a system in which every device in the group is paired with every other device, the effort required is reduced from $O(N^2)$ to $O(N)$. Saving the user time and effort is not the only merit of CPP, however, as it also enables the pairing of two mobile devices without a common OOB channel. For example, a wireless headset can not be paired using the barcode-based OOB channel, which requires a camera and a display. Assume that the laptop also is not able to communicate using the gesture-based OOB channel. Consequently,

Bob can use the smart phone, which can pair using both barcode-based and gesture-based OOB channels, as a bridge between the laptop and the headset.

Although CPP allows pairing of two mobile devices, this result shows the essential features of SGA. Once CPP is completed successfully, more than two mobile devices can possess the authenticated public keys of other devices. In this situation, they can easily build a secure group at any time.

3.4.2 Centralized Model

Proposed accumulative SGA method assumes that every mobile device is able to pair using at least one OOB channel. However, even if this requirement is satisfied, establishing a fully authenticated group may be impossible in some cases. For example, suppose Bob has seven devices and he wants to associate all of them. All possible pairings are performed and the certificates are issued, as shown in Figure 3.3.



Figure 3.3: Limitations in the accumulative SGA method

Consider the following cases:

- *No matching OOB channel*: For example, Bobs camera can pair with other devices only using an LED-based OOB channel [24]; however, no other devices can communicate using this channel.
- *No direct found*: Even if there is a certification path, for example, from the iPod to the headset, CPP may not occur between them directly because the certification path is supposed to include two certificates only: one is the requester received from the middle device and another is the verifier issued to the middle device in the proposed method. Therefore, in order to execute CPP between iPod and the headset that has the certification path of three certificates, CPP

is carried out first either between the headset and the laptop or between the iPod and the phone.

- *No path at all*: CPP may never be executed between some pairs of devices due to the lack of the certification path between them. If two mobile devices share the same OOB channel, they are able to be paired using the user-aided authentication. In Bobs case, the printer is paired with the iPad only and the laptop is paired with the phone and iPod. However, there is no certification path found between the printer and the laptop.

The centralized model of the method can address these limitations. To build the centralized model, one of the mobile devices must be designated as a groupHub. This should be the device with the greatest computing power and the most user interfaces. In Bob's case, his laptop can be the groupHub. All OOB channels that are to be used must be installed in the groupHub, whereas only one OOB channel is required for the other mobile devices.

In centralized model, the accumulative secure group association proceeds as follows. First, each device is paired with the groupHub using the desired OOB channel, and certificates are issued. Following this, every pair of devices can automatically be paired using CPP, as shown in Figure 3.4. A device can join the group at any time by following these steps.



Figure 3.4: Centralized model of the accumulative SGA method

3.4.3 Certification Path Pairing (CPP) Protocol

In the proposed method, a secure group is established in an accumulative manner. To join the group, a new member pairs with the groupHub, first, using their desired OOB channel. After the pairing is successfully completed and certificates are exchanged, the new member is supposed to pair with other group members at any time using the certification path. Therefore, we have proposed Certification Path Pairing (CPP) protocol. CPP protocol is designed in such a way that the new member can mutually associate one or more existing group members simultaneously, as depicted in Figure 3.5.

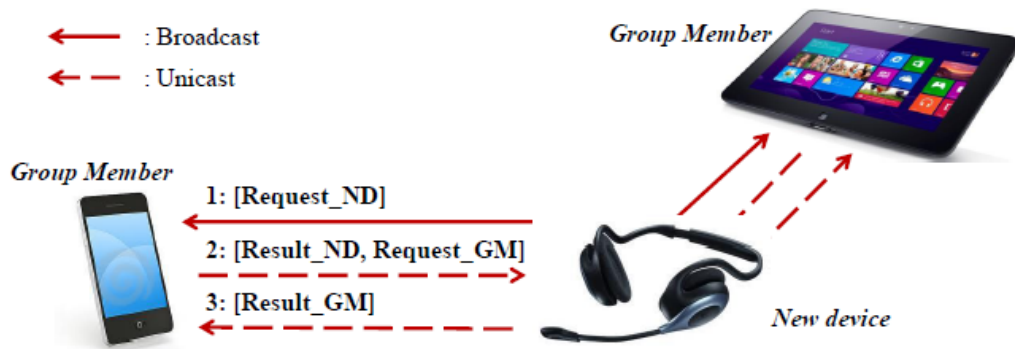


Figure 3.5: CPP protocol in general

In CPP protocol, three rounds are performed between the new device and every group member to certify mutually. The new device initiates the protocol by broadcasting a request (Request_ND). Once the group member receives the request, it establishes a regular unicast connection with the new device, and sends the result of the request (Result_ND) along with its own request (Request_GM). In the end, the new device sends the result of the received request (Result_GM) back to the group member.

Figure 3.6 depicts the sequence of the first half (Request_ND and Result_ND) of the CPP protocol as another half (Request_GM and Result_GM) is the exactly same. ndCont and ndStore represent objects on the new device, and gmCont and gmStore are on the group member side.

The rounds of the protocol are expanded on as follows.

ROUND 1. The new group member broadcast a request, which includes the certificate sender request (aCSR) and the certificates that were issued by the groupHubs (hubCert []). Mobile devices may join many different groups, thus there may

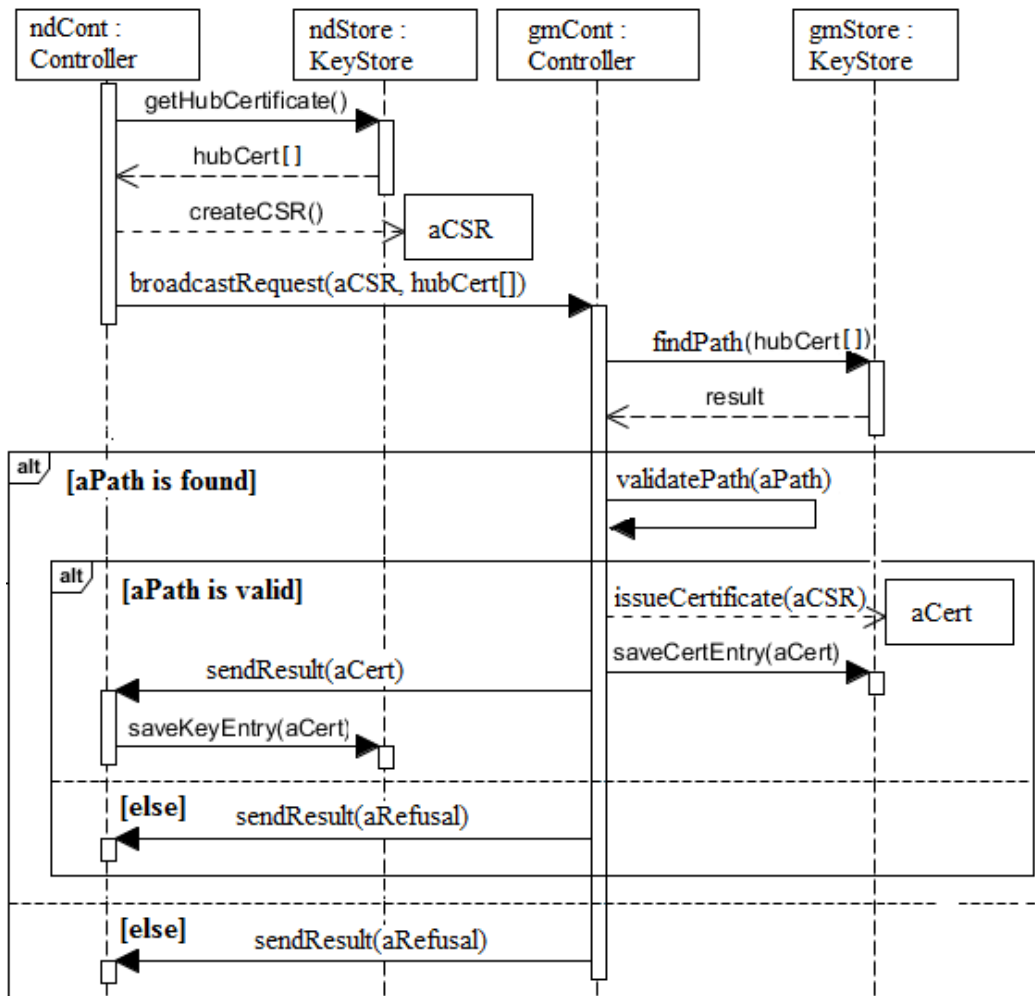


Figure 3.6: CPP protocol sequence

be more than one groupHub certificates in their KeyStore. It does not matter which groupHub certificate is used as long as there is certification path between two devices.

ROUND 2. Round 2: After receiving the request, group member searches a certification path to the new device (findPath). The certification path is supposed to consist of two certificates: one is groupHubs issued certificate to the new device and another is the group members issued certificate to the groupHub. The new device sends groupHubs issued certificate, and therefore, to find the path, the group member tries to find the certificate, of which the subject is the equal

to the issuer of the received certificate, from its Keystore. If the path exists, it needs to be validated (`validatePath`). To do that, expiration dates of both certificates are checked first. Then, the digital signature of the received groupHub certificate is verified by using its public key on certificate of the group member. If the path is valid, group member then issues a certificate (`aCert`) to the new device and saves it in its Keystore as a certificate entry (`saveCertEntry`). Otherwise, it refuses to certify. Therefore, the result includes either the refusal or the newly issued certificate. The new device also saves the received certificate in its own Keystore as a key entry (`saveKeyEntry`).

3.4.4 Role of the groupHub

If the groupHub certifies all of the group members, why do we need the mutual certification between each pair of group members? In the proposed method, the groupHub is not a trusted third party and does not take on the same level of responsibility of a certificate authority. Instead, it serves only as a bridge between other devices to enable authentication. Moreover, relying on one mobile device completely ubiquitous environment should be avoided. In other words, group association should be as flexible as possible and has the ability to move on in any cases. Therefore, if there is no sign of the groupHub when adding a new device, a different device in the group can be designated as the groupHub. In that case, the new device is able to join the group as following the same steps as usual because the new groupHub has already exchanged certificates with other group members. That means that the group can have more than one groupHub.

Moreover, a mobile device can join more than one group, and remain for a long period of time, due to the ad-hoc nature of mobile networks. For example, Bob can create a secure group for his personal devices at home. Furthermore, his laptop may also be a member of another group in his office. This means the device can possess several groupHub certificates. Selecting a specific group is not possible in CPP protocol. However, the proximity of mobile devices may assist to distinguish the groups. For example, when Bob starts CPP protocol with his laptop in the office, Alice's device is most likely to find a certification path with the groupHub of the office.

3.5 Prototype System

3.5.1 System Setup

For a comparative experiment, we implemented two prototype systems: our proposed accumulative SGA method and an all-at-once association. The purpose of the experiment is to compare the mean completion time of two associations.

We chose SAS-GMA [75] protocol as a representative of the all-at-once association to compare with our accumulative SGA method for a couple of reasons. First, as presented in Table 3.1, it is one of the protocols that require the minimum user involvement. To associate n mobile devices, SAS-GMA protocol only needs $(n-1)$ OOB channel transfers. Without verifying the size of the group, all-at-once association protocols could not prevent intruders from joining. Thus, it is a serious drawback of SAS-GMA protocol. If size verification is added to the protocol, its completion time will notably increase.

Second, unlike other group association protocols, SAS-GMA does not specify a type of OOB channel that it needs. However, it has to be the same or similar for all group members, so that SAS authentication data can be verified simultaneously. We decided to use a barcode-based OOB channel for SAS-GMA implementation to avoid the human related errors, because the completion time of the successful associations is needed for our experiment. In d2d type of OOB channels, including the barcode-based, the users have relatively little influence on the success of the transmission. Those channels transfer the authentication data directly from the requester to the verifier, and the users only assist in making the transmission possible. Furthermore, smart phones that have both a screen and a camera are used in our experiment, and therefore, a barcode-based transfer was the handiest in our situation. To equalize with SAS-GMA, the accumulative SGA method also utilizes the barcode-based OOB channel for its pairings.

Both prototype systems were developed on smart phones running the Android operating system. The implementation used Java language and cryptography part was written using Bouncy Castle library [92], which contains a lightweight cryptography API suitable for memory-constrained devices. The Barcode-based OOB channel was implemented using ZXing open-source library [93]. The communication between devices used WLAN. Table 3.3 shows the technical specification of the smart phone handsets that we used in the experiment.

Table 3.3: Technical specification of handsets

Device	CPU	RAM	Android Version
Samsung Galaxy Note	Cortex-A9 dual-core 1.4 GHz	1GB	4.0 Ice Cream Sandwich
Samsung Nexus S	Cortex-A8 dual-core 1.5GHz	1GB	4.0 Ice Cream Sandwich
Samsung Galaxy 2S	Qualcomm Scorpion dual-core 1.5GHz	1GB	2.3 Gingerbread
Pantech Mirach	Qualcomm Snapdragon dual-core 1GHz	512MB	2.3 Gingerbread
Sharp IS01	Qualcomm QSD8650 1GHz	256MB	1.6 Donut

Table 3.4: Computational overhead of cryptographic operations (msec)

Operation	Galaxy Note		Nexus S		Galaxy 2S		Mirach		IS01	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
RSA Key Generation	620.2	285.18	799.3	472.83	625.78	328.03	910.2	614.82	804.4	410.24
CSR Generation	57.92	18.06	35.4	13.96	30.16	36.86	36.84	6.63	48.3	35.75
Certificate Verification	69.18	15.1	34.28	5.67	25.5	1.31	30.84	1.26	50.66	34.91
Certificate Generation	37.56	6.82	94.4	36.49	93	43.19	104.5	35.89	297.28	209.21

Table 3.4 presents the computational overhead of the handsets for several cryptographic operations. Each Data is summarized from 10 trials. RSA operations are performed using 1024-bit keys taking into account low-power mobile devices. As can be seen from the table, the key generation consumes the most time; however, the key pair is only created when the system runs for the first time.

3.5.2 Proposed Method Implementation

Our proposed accumulative SGA method consists of two parts. To join a group, a new device pairs with the groupHub first, and then it associates other group members using CPP protocol, explained in Section 3.4.3. For the first part, the prototype system adopted modified SAS protocol [23], as it can authenticate the public keys of both devices with one OOB channel transfer. As a result, the number of barcode-based OOB channel transfer becomes $n-1$, where n is the size of the group. The sequence of the modified SAS protocol

in our implementation is depicted in Figure 3.7. The new device initiates the protocol by broadcasting a request.

The protocol consists of six rounds, including the OOB channel transfer. Even though not shown on the figure, every transmission between devices has a unique identifier to protect from replay attack. The rounds of the protocol are expanded on as follows.

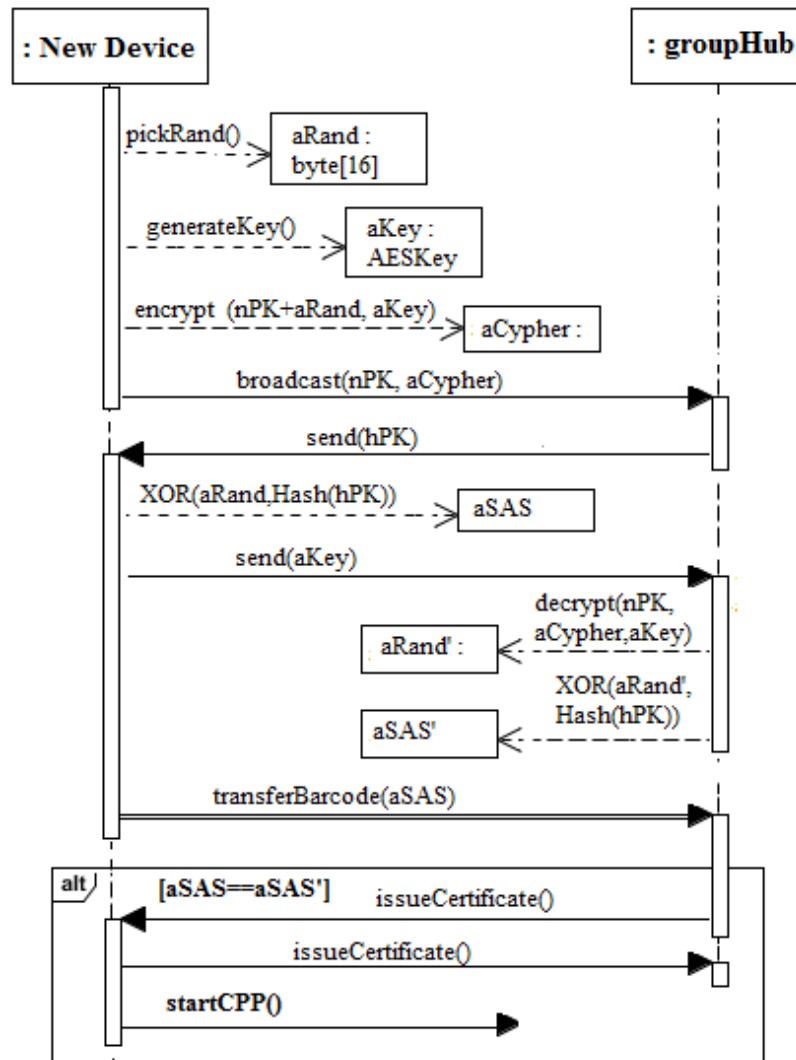


Figure 3.7: Sequence of the modified SAS protocol in the prototype

ROUND 1. Primary purpose of the SAS protocol [14] is to reduce the length of the authentication data, so that the low-bandwidth OOB channels can be used. However, our selected barcode-based OOB channel is capable of transferring data that is

bigger than the size of public key. Thus, the prototype uses 16 byte SAS data instead of 16 bit (pickRand). Moreover, a symmetric cryptography function (generateKey) is used for the implementation of the commitment scheme. The encrypt() function replaces the 'COMMIT' phase and the decrypt() function replaces the 'OPEN' phase of the commitment scheme. Thus, the new device encrypts its public key (nPK) as well as the random (aRand) with the generated key (aKey) and broadcasts its public key (nPK) along with the encrypted data (aCypher).

- ROUND 2. When the groupHub receives the request, it establishes a regular unicast connection with the new device, and sends its public key (hPK). Figure shows the public keys only in round 1 and round 2, for the sake of the simplicity. However, in the actual prototype system, a certificate sender request (CSR), which includes not only the public key but also all necessary information about the device for the certification, is sent.
- ROUND 3. Once the new device receives the public key of the groupHub, it sends back the symmetric key (aKey). The groupHub then decrypts the received data in round 1 with the key to get the committed value of the new device. Finally, both devices compute the SAS authenticatin data (aSAS) independently by performing exclusive OR operation between the committed value (aRand) and the hash of the public key of the groupHub.
- ROUND 4. BARCODE transfer takes place with the assistance of the user. The new device converts own SAS data into the QR code, and then displays it on its screen. The groupHub scans the code using its camera.
- ROUND 5. The groupHub converts the scanned QR code into the SAS data. If the received SAS data through the barcode transfer is the exactly same as its own computed SAS data, the groupHub issues a certificate to the new device. Otherwise, the groupHub sends the refusal.
- ROUND 6. If the new device receives the certificate from the groupHub, it also issues a certificate and sends it to the groupHub.

On completion of the successful pairing with the groupHub, the new device launches the CPP protocol by broadcasting a request in order to associate with other group members. CPP protocol consists of the additional three rounds, as shown in Figure 3.5.

3.5.3 SAS-GMA Protocol Implementation

SAS-GMA protocol is very similar to the SAS pairing protocol. Main difference is that there is no unicast connection between devices. Instead, all group members simultaneously communicate through broadcasting as shown in Figure 3.8. It has five rounds including a barcode-based OOB channel transfer. Even though not shown on the figure, every transmission between devices has a unique identifier to protect from replay attack. Rounds of the protocol are expanded on as follows.

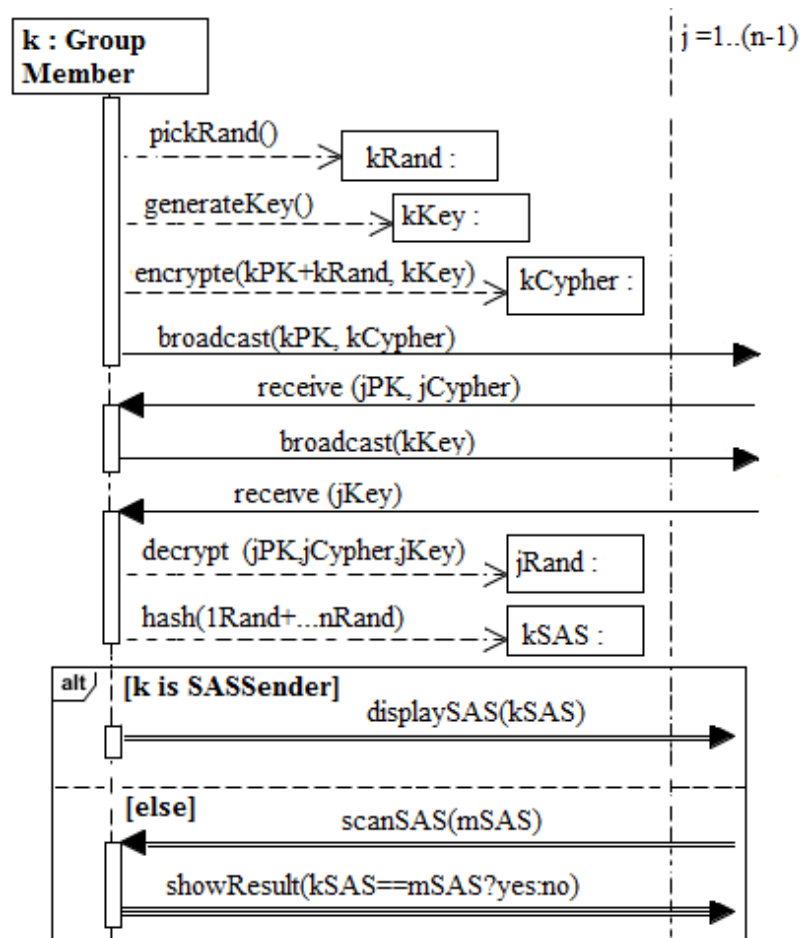


Figure 3.8: Sequence of the SAS-GMA protocol in the prototype

ROUND 1. Every device broadcasts a request that includes their public key and encrypted data simultaneously. The procedure of committing the random value is the

same as the one, described in round 1 of the accumulative SGA method implementation.

ROUND 2. Every device receives the requests of other group members. Round 1 and round 2 occur synchronously. SAS-GMA protocol is designed for devices to wait for a while before moving to the next round. An initial version of the prototype was implemented as it is. Because our experiment considers small groups with size of 4 to 6 members, the prototype was made to wait for 3 seconds. Time is measured from the device sending its request. However, the main goal of the experiment is to compare the mean completion time of two types of associations. If one system stops for seconds of time during the association, another system will get the benefit from it. Therefore, we decided to change the protocol slightly, and in our final version of the prototype, the participants input the expected group size into the system. As a result, instead of waiting, the system proceeds to the next round when it receives the same number of the requests as the group size.

ROUND 3. Every device broadcasts their encryption key to open their commitments.

ROUND 4. Every device receives the encryption keys of the other group members. Round 3 and round 4 occur synchronously. Once the device receives the keys from all devices that have sent the request, it decrypts all received data in round 1. Every device then independently computes the SAS authentication data. To do that, all committed values must be put into an order by their senders identities. Our prototype sorts them based on IP address of the device. Then, we concatenate the sorted values because the barcode-based OOB channel is not low-bandwidth, and so the length of the SAS data is not a problem in the prototype. Finally, the SAS data is ready by hashing the concatenated value ($hash(1Rand + \dots + nRand)$).

ROUND 5. BARCODE transfer takes place with the assistance of the users. Instead of transferring the SAS data between every pair of devices, the prototype chooses one device (SASsender) to display the SAS data as the form of a QR code (displaySAS) and other members scan it by their camera (scanSAS). It reduces the number of OOB channel transfer from $n * (n - 1)/2$ to $n-1$, where n is the size of the group. After scanning the code, the device checks the received SAS

data against its own created one, and then informs the result (showResult).
The association accomplishes, if all devices verify the SAS data.

3.5.4 User Interface

Figure 3.9 presents the user interface of the prototype system of the proposed method. Interface (a) is the main interface of the experiment. To add their device to a group, the participants press the ‘Join Group’ button. ‘Reset’ button is to prepare the device for the next experiment. It also informs the participant in real-time of how many certificates are in the keystore. Interface (b) shows the basic information about the device. The prototype system gives a unique ID, which starts with ‘SMG’, to each device to avoid relying on the specific hardware, and it represents the device during the association. The groupHub can be designated on this settings as well. Interface (c) reads the keystore and shows both type of certificates.

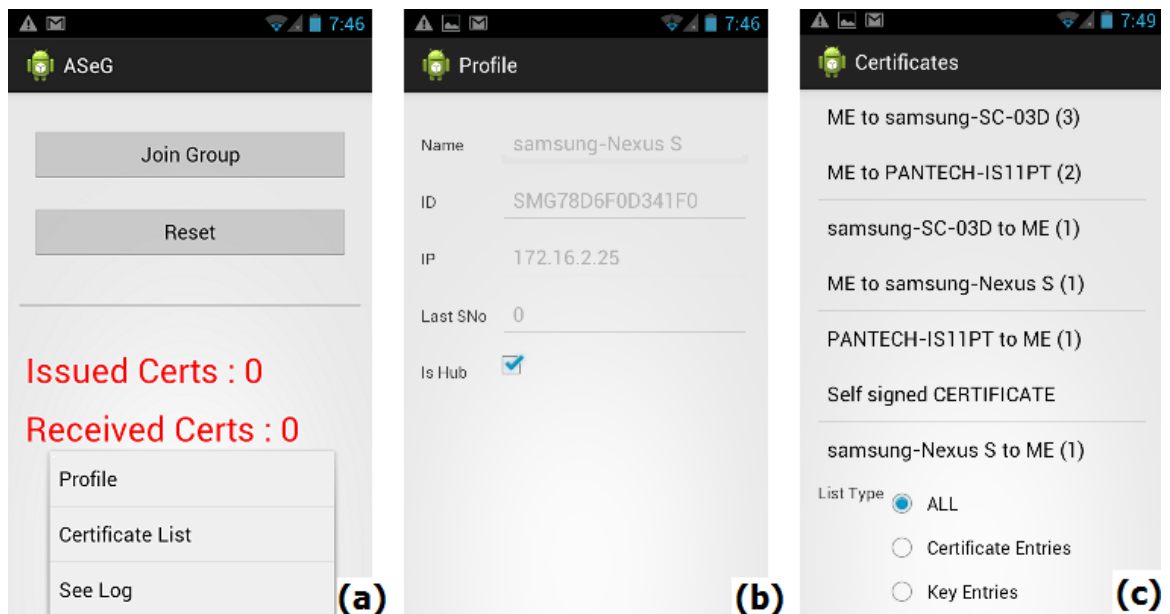


Figure 3.9: User interface of the proposed method: (a) experiment (b) settings (c) Keystore reading

Figure 3.10 presents the user interface of the prototype system of SAS-GMA protocol. Interface (a) is the main interface of the experiment. To associate devices, the participants presses ‘Start’ button on their respective devices simultaneously. Once the association completed, it displays the result of the SAS data verification. During the association, it



Figure 3.10: User interface of the SAS-GMA protocol: (a) experiment (b) settings (c) barcode transfer

informs the participant in real-time of how many devices it is interacting with. Interface (b) is similar to that of the previous prototype and the SASSender is designated here. Moreover, the group size is chosen prior to each association on this settings. Interface (c) shows how the SASSender displays the converted QR code along with the original SAS data.

3.6 Comparative User Experiment

3.6.1 Purpose

In this experiment, we have investigated a comparative study of the one-by-one and all-at-once group association methods in terms of the completion time. One-by-one association has a number of advantages over all-at-once associations, as reported in Section 3.3. It has stronger scalability, greater flexibility, unlimited size, better device diversity, and etc. However, one-by-one association is intuitively considered that it needs much more time to complete. Therefore, the purpose of this experiment is to clarify this particular assumption. The hypothesis of the study is that one-by-one association is not slower than all-at-once association.

3.6.2 Tasks

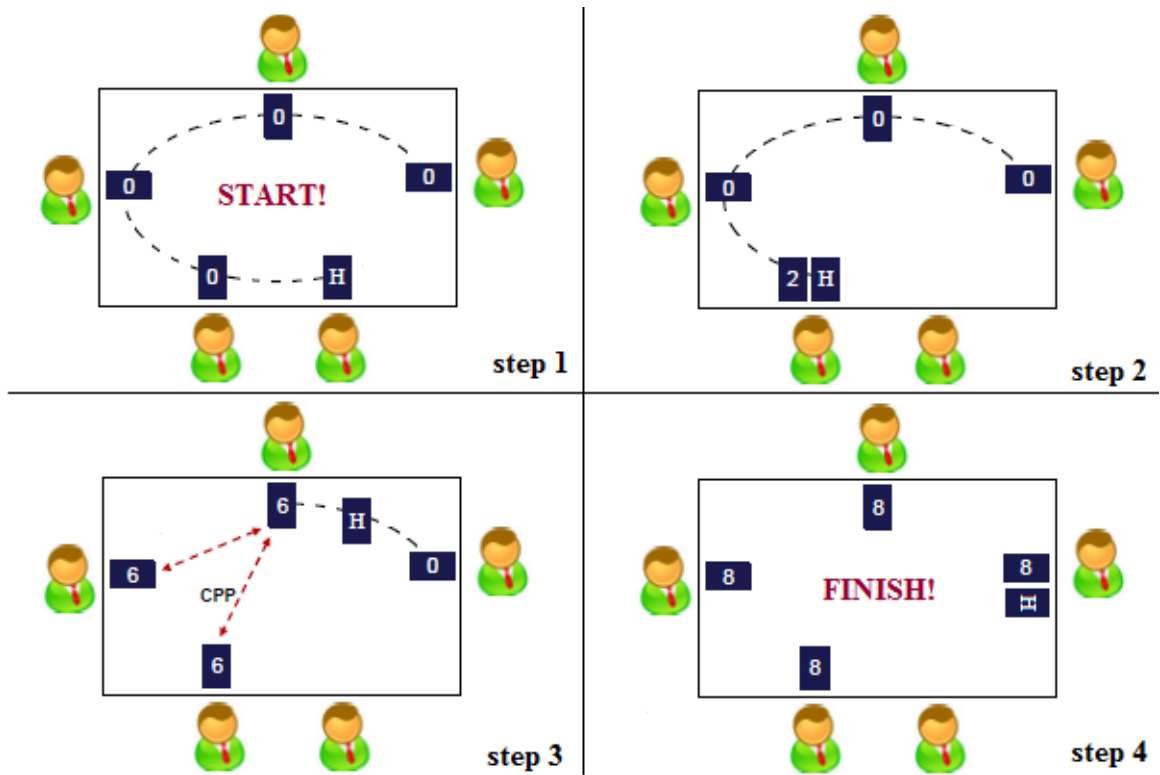


Figure 3.11: Task sequence of the proposed method: Completion time is measured from starting the first member pairing with the groupHub to completing CPP protocol of the last member.

Figure 3.11 depicts the task sequence of the experiment that associates mobile devices using our accumulative SGA method. Before an association, one device is designated as the groupHub (marked as 'H' in the figure); however, time for this action is not counted in the total association time. At this point, a group has no member, and so devices have no certificates (step 1). Participants then hold the groupHub in turn to pair it with their respective devices using SAS protocol with the barcode-based OOB channel. If the pairing is successful, both devices get two more certificates: a received certificate and an issued certificate (step 2). As soon as the pairing is completed, the newly joined device starts pairing again with other group members simultaneously using CPP protocol. Each pairing also adds two more certificates to its corresponding devices (step 3). When the association is completed, each group member is supposed to possess $2 * (n - 1)$ certificates, where n is

the number of devices (step 4). The total association time is measured from starting the first member pairing with the groupHub to completing CPP protocol of the last member.

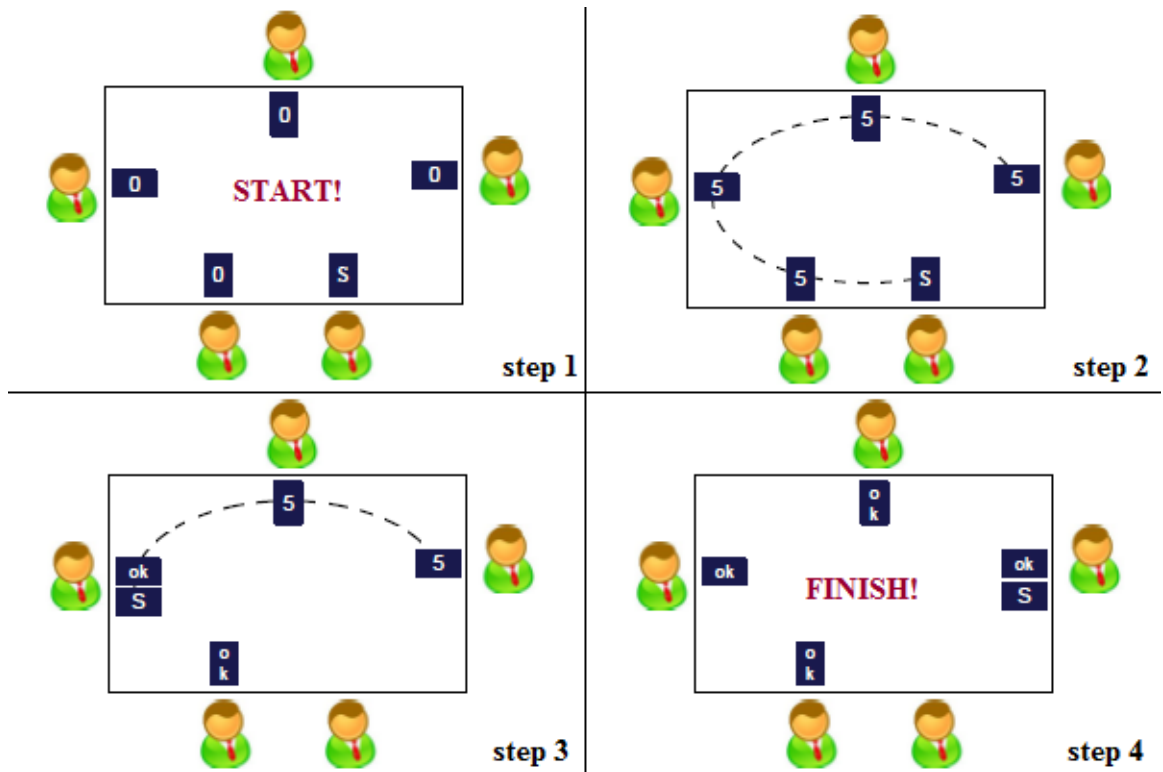


Figure 3.12: Task sequence of the SAS-GMA protocol: Completion time is measured from launching association on all devices to ending the last device displaying the result of the verification.

Similarly, Figure 3.12 depicts the task sequence of the experiment that associates mobile devices using SAS-GMA protocol. Before an association, one device is designated as a SASSender (marked as ‘S’ in the figure). In addition, participants input to their respective devices how many devices would be associated by selecting the group size on the settings. However, time for these actions is not counted in the total association time (step 1). All participants then start the association with their respective devices at the same time. On completion of SAS-GMA protocol, each device displays how many members they have connected (step 2). Finally, participants hold the SASSender in turn to verify the computed SAS data of their devices using the barcode-based OOB channel. The device informs the result of the verification by displaying ‘OK’ or ‘Not matched.’ (step 3, step 4). The total

association time is measured from launching association on all devices to ending the last device displaying the result of the verification.

3.6.3 Participants

Fifteen volunteers (eight females and seven males) participated in our experiment; all of them were either undergraduate or graduate students. Their mean age was 29.7 years (SD=5.79 years, range 20-38). Before beginning the experiment, a brief introduction was given on how to carry out both associations.

In line with other group association studies [77, 79, 80], we considered a common setting with small groups (sizes 4-6). Various combinations of different handsets of which technical specifications are presented in Table 3.3, were tested. Participants were randomly selected into 4, 5, and 6 persons groups. Each group carried out both one-by-one and all-at-once associations. We have conducted total of ten associations for each method in all three group sizes. Therefore, a single participant has involved in more than one association, but no two groups were the exactly same. Figure 3.13 (a) illustrates how a group of five participants are performing associations in the experiment.



Figure 3.13: User experiment: (a) multiple user - 5 participants are discussing a task before the association (b) single user - a participant is associating 4 devices

In addition to the multiple user experiment, we have conducted the same experiment in a single user scenario. The aim of this experiment is to observe the completion time difference between one-by-one and all-at-once associations when a single person associates a group of mobile devices. Thus, each of fifteen participants was asked to carry out the

both group associations in three different group sizes alone. The task sequences were the basically same as those of multiple user cases. The only difference was that the participant did not need to pass the groupHub or SASSender to others, and as a result, the designated groupHub or SASSender devices became the center of their association processes. Figure 3.13 (b) illustrates how a participant is associating devices in the experiment.

3.6.4 Result

Table 3.6 summarizes the completion time of two methods, the proposed accumulative SGA method and SAS-GMA protocol, in three group sizes. We analyzed the collected data using paired tTest between two methods in each group size to determine if there are statistically significant differences. The results are presented on the middle column of the table, and they clearly demonstrate that no significant differences between the completion times of one-by-one and all-at-once associations ($p > 0.05$) were found for group size of 4, 5, and 6 devices.

Table 3.5: Completion time of associations - multiple users (sec)

Size	Proposed Method		Paired tTest (df=10)	SAS-GMA	
	Mean	SD		Mean	SD
4	27.05	3.01	t=0.92, $p=0.383$	25.29	4.55
5	34.32	2.68	t=1.38, $p=0.200$	32.18	3.35
6	42.11	3.22	t=-0.57, $p=0.581$	43.94	8.08

We also asked our participants to carry out the both group associations alone in order to observe the differences in single user cases. However, the results were very similar to those of the multiple user cases, as shown in Table 3.5.

Table 3.6: Completion time of associations - single user (sec)

Size	Proposed Method		Paired tTest (df=14)	SAS-GMA	
	Mean	SD		Mean	SD
4	26.70	4.72	t=-0.95, $p=0.359$	28.90	8.27
5	33.60	5.31	t=-1.09, $p=0.295$	35.83	6.01
6	42.87	8.27	t=0.16, $p=0.872$	42.39	5.11

In the end, we analyzed the same data again to see whether there are statistically significant differences between the completion time of multiple-user and single-user cases. Paired tTest was also used for each association method in all three group sizes. As shown in Table 3.7, no significant differences ($p > 0.05$) in all six cases were found.

Table 3.7: Comparison between multiple and single user associations

System	Paired tTest (df=9)		
	4	5	6
<i>Proposed method</i>	t=0.60, p=0.562	t=-1.85, p=0.097	t=-1.50, p=0.167
<i>SAS-GMA</i>	t=1.56, p=0.153	t=2.14, p=0.061	t=-0.59, p=0.57

As a result, it can be concluded that both one-by-one and all-at-once group associations spend the similar amount of time in the small group of less than 6 devices for both single and multiple user cases. Moreover, the number of users does not affect the completion time of the group associations.

3.7 Discussion

In our user experiment, we asked the participants to associate all devices immediately because the purpose of the study was to measure the association time. However, in general, the proposed accumulative SGA method is supposed to associate devices without any limitation on time and place. For instance, assume that 5 employers associated their personal devices during a meeting using our method. On the next scheduled meeting, an employer could not participate while two new comers joined. New employers could easily add their devices to the group even though one member was missing. The device of employer who was absent is able to associate with new group members in any place at any time using CPP protocol that is automatic.

The completion time of the user-aided authentication relies profoundly on the user involvement in the association process because the speed of OOB channel transfer is much slower than wireless communications no matter how slow the network is and how many rounds the protocol performs. Moreover, it varies depending on OOB channel type. We used a barcode-based OOB channel in our experiment for both of all-at-once and one-by-one associations. All-at-once association basically requires the same OOB channels for

all members. In contrast, our method is supposed to associate devices using their desired OOB channels. Therefore, the experiment we have conducted is not realistic to some extent. However, it has proved that a tendency to believe that the one-by-one association spends more time is wrong.

In general, the process of establishing a secure group consists of two parts: secure group association, also called trust establishment, and secure group formation (SGF), or key management. SGA and SGF should occur as independently as possible in ubiquitous environment. Due to the ad hoc nature of mobile networks, changes in the group tend to happen frequently, and so efficient handling is required. Our proposed method considers SGA only. To interact securely within the group, SGF method, which determines a group key, is necessary. Once the group members are authenticated successfully using the accumulative SGA method, some or all members can use their authenticated public keys to establish the group key using one of the existing SGF protocols [94–96]. This process should be fast and easy as it does not require user involvement.

To add a new member is easy in the proposed accumulative SGA method. To do that, the new member pairs with groupHub first, and then it associates with other group members using CPP protocol. However, how to remove the member from the group is not a simple task and must be considered carefully. A successful execution of the proposed method exchanges digital certificates among group members; therefore the system needs to revoke certificates to remove the undesired member. In an ad-hoc and ubiquitous network, distributing a certificate revocation list is not practical. The simplest solution to this problem is to adjust certificate lifetimes. In addition, the group can exclude the undesired member from the group key establishment protocol.

The digital certificate in the proposed accumulative SGA method binds the public key to a unique hardware identifier of the mobile device, such as MAC address. However, for some cases, it may need to include a user-friendly name of the device. The model number or manufacturers name of the device cannot be used, because these do not provide a unique identification. Names that can be changed by the user such as a Bluetooth-friendly name can serve this purpose. However, it has been reported that 18% of users do not change the Bluetooth-friendly name of their device from manufacturer default [97]. Moreover, some devices may not have an interface that allows the user to change the device name. Therefore, to address this problem, the user may input the name into the device using the groupHub

by including it in the groupHub certificate during the pairing because the groupHub is expected to have rich user interface.

3.8 Summary

This chapter introduced our proposed accumulative SGA method that utilizes various types OOB channels. There has been little discussion of setting up secure connections between groups of mobile devices. Some security protocols have been proposed, but they have tended to focus on a scenario whereby all devices must be physically located in the same place to perform the association. In contrast, our method is designed for a broader range of scenario and does not require all devices to be in one place at one time. It occurs in an accumulative manner, allowing each device to join the group independently from the other group members and their associations. Our proposed method assumes that all mobile devices are capable of pairing using at least one OOB channel. To join a group, a device must perform OOB channel pairing with one of the group members. Digital certificates are then issued upon the successful authentication. Once certificates are exchanged between the new device and one member, pairing with other group members becomes automatic owing to the certification path.

We have implemented a prototype system and conducted a comparative user experiment in order to ensure viability of the proposed method. The result proved that the accumulative association spent the same amount time as the simultaneous group associations. In addition, we reported comparison on accumulative and simultaneous associations and it clearly demonstrated our proposed method has a number of attractive features, such as stronger scalability, greater flexibility, unlimited group size, better device diversity and etc, over the existing secure group association protocols.

Chapter 4

Conclusion

The increased popularity of mobile devices, such as laptops, mobile phones, tablets, accessory devices, and more, brings new challenges in the area of network security. In ubiquitous environment, where ad-hoc networks may be formed using mobile devices, virtual connection or pairing is required before transferring any data wirelessly. Large numbers of researches have focused on the user-aided authentication to establish a secure connection between mobile devices, commonly referred as *secure device pairing*. The user-aided authentication is an authentication technique that is generally based on a human perceivable channel, called out-of-band (OOB) channel, through which the authentication data can be transferred.

A significant number of OOB channels have been proposed so far. The principle limitation of these approaches, however, is that every one of them is devoted to one particular situation and is not intended to be used for various mobile devices. In general, a universally applicable OOB channel is unlikely to exist in the near future due to the different capabilities of a wide variety of mobile devices as well as human users who have diverse preferences. Therefore, if a mobile device possesses as many as possible OOB channels, its possibility of successful pairing in ad-hoc and ubiquitous environment will increase. Based on this concept, we have defined our research vision, “the more OOB channels, the more security”.

4.1 Contributions

To support the research vision, we have proposed two new methods.

First, we have introduced a new OOB channel that uses an accelerometer-based gesture input. Gesture-based OOB channel is suitable for all-kinds of mobile devices, including input/output constrained devices, as the accelerometer is small and incurs only a small computational overhead. In our OOB channel, one device converts the authentication data into a sequence of gestures and informs a user of them. The user then performs the gestures one-by-one with the second device that has an embedded accelerometer to transfer the data into it. We have implemented a prototype system and conducted thorough usability analysis to compare our gesture-based OOB channel with the existing OOB channels. Result showed that the gesture-based OOB channel has reliable performance with a mean completion time of 16.86 seconds with $SD=4.2$ seconds, as well as a mean error rate of 0.13 per transfer with $SD = 0.34$. Four different gesture sets were used in this experiment to determine which library size would be more usable in practice. The result suggested that library size of 8 gestures was the most effective among our tested sizes. We have also conducted another experiment to determine whether user-defined gesture templates improve the accuracy of gesture recognition. Contrary to expectations, this study did not find a significant difference between the predefined and user-defined templates in terms of the error rate.

Second, we have designed a new secure group association method that utilizes various types of OOB channels. The term *secure group association* is generally understood to mean that every device pairs with other group members. There has been little discussion of setting up secure connections among group of mobile devices. Some security protocols have been proposed, but they have tended to focus on a scenario whereby all devices must be physically located in the same place to perform the association. In contrast, our method is designed for a broader range of scenario and does not require all devices to be in one place at one time. It occurs in an accumulative manner, allowing each device to join the group independently from the other group members and their associations. To join a group, a device must perform secure pairing with one of the group members. Digital certificates are then issued upon the successful authentication. Once certificates are exchanged between the new device and the member, pairing with other group members becomes automatic owing to the certification path. We have implemented a prototype system and conducted a comparative user experiment in order to ensure viability of the proposed method. The result proved that the accumulative group association spends the same amount time as the simultaneous group association. In addition, we have reported comparison on accumulative and simultaneous associations and it clearly demonstrates our proposed method has a num-

ber of attractive features, such as stronger scalability, greater flexibility, unlimited group size, better device diversity and etc, over the existing protocols.

There will be more portable devices and more collaborative work in the near future, and the user-aided authentication will still play an important role to secure ad-hoc and ubiquitous networks. We hope that our proposed methods contribute to build an secure environment, where the security is always available on demand.

4.2 Future Work

In the future, we would like to implement a complete library that is based on our accumulative secure group association method. The library will include all kinds of secure pairing protocols, all types of OOB channels, as well as alternative wireless technologies, so that any mobile device is able to pair or associate in the ubiquitous environment regardless of their physical capabilities. We want to investigate other security technologies, such as TinyECC [98], which are more suitable for the resource constraint mobile devices, and also can substitute expensive public key cryptography operations while offering the same degree of security and functionality. Furthermore, we need to study social factors of the users during the group association.

Bibliography

- [1] F. Stajano and R. Anderson, “The resurrecting duckling: Security issues for ad-hoc wireless networks,” in *Proceedings of the 7th international workshop on Security Protocols*. London, UK: Springer-Verlag, 1999, pp. 172–182.
- [2] M. A. Hanson, H. C. Powell, A. T. Barth, K. Ringgenberg, B. H. Calhoun, J. H. Aylor, and J. Lach, “Body area sensor networks: Challenges and opportunities,” *IEEE Computer*, vol. 42, no. 1, pp. 58–65, 2009.
- [3] C. Adams and S. Lloyd, *Understanding PKI: Concepts, Standards and Deployment Considerations*. Addison Wesley, 2002.
- [4] D. Balfanz, D. K. Smetters, P. Stewart, and H. C. Wong, “Talking to strangers: Authentication in ad-hoc wireless networks,” in *Proceedings of the symposium on Network and Distributed System Security*, ser. NDSS’02. Virginia, USA: Internet Society, 2002, p. 13.
- [5] V. Boyko, P. MacKenzie, and S. Patel, “Provably secure password-authenticated key exchange using deffie-hellman,” in *Proceedings of the International Conference on Advances in Cryptology*, ser. EUROCRYPT’00. Berlin, Heidelberg: Springer-Verlag, 2000, pp. 156–171.
- [6] M. Cagalj, S. Capkun, and J. Hubaux, “Key agreement in peer-to-peer wireless networks,” *Proceedings of the IEEE: Special Issue on Security and Cryptography*, vol. 94, no. 2, pp. 467–478, 2006.
- [7] S. Creese, M. Goldsmith, R. Harrison, B. Roscoe, P. Whittaker, and I. Zakiuddin, “Exploiting empirical engagement in authentication protocol design,” in *Proceedings*

- of the 2nd International Conference on Security in Pervasive Computing*, ser. SPC'05. Berlin, Heidelberg: Springer-Verlag, 2005, pp. 119–133.
- [8] C. Gehrman, C. Mitchell, and K. Nyberg, “Manual authentication for wireless devices,” *RSA Cryptobytes*, vol. 7, no. 1, pp. 29–37, 2004.
- [9] J. H. Hoepman, “Ephemeral pairing on anonymous networks,” in *Proceedings of the 2nd International Conference on Security in Pervasive Computing*, ser. SPC'05. Berlin, Heidelberg: Springer-Verlag, 2004, pp. 101–116.
- [10] S. Laur and K. Nyberg, “Efficient mutual data authentication using manually authenticated strings,” in *Proceedings of the 5th International Conference on Cryptology and Network Security*, ser. CANS'06. Berlin, Heidelberg: Springer-Verlag, 2006, pp. 90–107.
- [11] L. H. Nguyen and A. W. Roscoe, “Efficient group authentication protocol based on human interaction,” in *Proceedings of the workshop on Foundation of Computer Security and Automated Reasoning for Security Protocol Analysis*, ser. FCS-ARSPA'06, 2006, pp. 9–31.
- [12] J. Valkonen, N. Asokan, and K. Nyberg, “Ad hoc security associations for groups,” in *Proceedings of the 3rd European Workshop on Security and Privacy in Ad-hoc and Sensor Networks*, ser. ESAS'06. Berlin, Heidelberg: Springer-Verlag, 2006, pp. 150–164.
- [13] F. L. Wong and F. Stajano, “Multi-channel security protocols,” *IEEE Pervasive Computing*, vol. 6, no. 4, pp. 31–39, 2007.
- [14] S. Vaudenay, “Secure communications over insecure channels based on short authenticated strings,” in *Proceedings of the 25th International Conference on Cryptology*, ser. CRYPTO'05. Berlin, Heidelberg: Springer-Verlag, 2005, pp. 309–326.
- [15] H. Delfs and H. Knebl, *Introduction to Cryptography: Principles and Applications*. Springer, 2002, ch. Cryptographic Protocols.
- [16] R. Dhamija and A. Perrig, “Deja vu: a user study using images for authentication,” in *Proceedings of the 9th conference on USENIX Security Symposium*, ser. SSYM'00. New York, NY, USA: ACM, 2000, pp. 4:1–4:14.

-
- [17] C. M. Ellison and S. Dohrmann, “Public-key support for group collaboration,” *ACM Transactions on Information and System Security*, vol. 6, no. 4, pp. 547–565, 2003.
- [18] M. T. Goodrich, M. Sirivianos, J. Solis, G. Tsudik, and E. Uzun, “Loud and clear: Human-verifiable authentication based on audio,” in *Proceedings of the 26th IEEE Conference on Distributed Computing Systems*, ser. ICDCS’06. Washington, DC, USA: IEEE Computer Society, 2006, p. 10.
- [19] J. M. McCune and A. Perrig, “Seeing-is-believing: Using camera phones for human-verifiable authentication,” *International Journal of Security and Networks*, vol. 4, no. 1/2, pp. 43–55, 2009.
- [20] A. Perrig and D. Song, “Hash visualization: A new technique to improve real-world security,” in *Proceedings of the International Workshop on Cryptographic Techniques and E-Commerce*, 1999, pp. 131–138.
- [21] V. Roth, W. Polak, E. Rieffel, and T. Turner, “Simple and effective defence against evil twin access points,” in *Proceedings of the 1st Conference on Wireless Network Security*, ser. WiSec’08. New York, NY, USA: ACM, 2008, pp. 220–235.
- [22] E. Uzun, K. Karvonen, and N. Asokan, “Usability analysis of secure pairing methods,” in *Proceedings of the International Workshop on Usable Security*, ser. USEC’07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 307–324.
- [23] N. Saxena, J. Ekberg, K. Kostianen, and N. Asokan, “Secure device pairing based on a visual channel,” in *Proceedings of the IEEE Symposium on Security and Privacy*. Washington, DC, USA: IEEE Computer Society, 2006, p. 17.
- [24] N. Saxena and M. B. Uddin, “Automated device pairing for asymmetric pairing scenarios,” in *Proceedings of the 10th International Conference on Information and Communication Security*, ser. ICICS’08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 311–327.
- [25] N. Saxena, M. B. Uddin, and J. Voris, “Universal device pairing using an auxiliary device,” in *Proceedings of the 4th International Symposium on Usable Privacy and Security*, ser. SOUPS’08. New York, NY, USA: ACM, 2008, pp. 56–67.

- [26] R. Prasad and N. Saxena, “Efficient device pairing using human-comparable synchronized audiovisual pattern,” in *Proceedings of the 6th International Conference on Applied Cryptography and Network Security*, ser. ACNS’08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 328–345.
- [27] M. T. Goodrich, M. Sirivianos, J. Solis, C. Soriente, G. Tsudik, and E. Uzun, “Using audio in secure devcie pairing,” *International Journal of Security and Networks*, vol. 4, no. 1/2, pp. 57–68, 2009.
- [28] C. Soriente, G. Tsudik, and E. Uzun, “BEDA: Button enabled device association,” in *Proceedings of the 1st International Workshop on Security for Spontaneous Interaction*, ser. IWSSI’07, 2007, p. 7.
- [29] —, “HAPADEP: Human-assisted pure audio device pairing,” in *Proceedings of the 11th International Conference on Information Security*, ser. ISC’08. New York, NY, USA: ACM, 2008, pp. 385–400.
- [30] W. R. Claycomb and D. Shin, “Secure device pairing using audio,” in *Proceedings of the 43rd Annual International Carnahan Conference on Security Technology*. IEEE, 2009, pp. 77–84.
- [31] A. Varshavsky, A. Scannell, A. LaMarca, and E. D. Lara, “Amigo: Proximity-based authentication of mobile devices,” in *Proceedings of the 9th International Conference on Ubiquitous Computing*, ser. UbiComp’07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 253–270.
- [32] R. Kindberg and K. Zhang, “Validating and securing spantaneous associations between wireless devices,” in *Proceedings of the International Conference on Information Security*, ser. ISC’03. Berlin, Heidelberg: Springer-Verlag, 2003, pp. 44–53.
- [33] I. Buhan, B. Boom, J. Doumen, P. Hartel, and R. Veldhuis, “Secure pairing with biometric,” *International Journal of Security and Networks*, vol. 4, no. 1/2, pp. 27–42, 2009.
- [34] R. Mayrhofer and M. Welch, “A human verifiable authentication protocol using visible laser light,” in *Proceedings of the International Conference on Availability, Reliability*

- and Security*, ser. ARES'07. Washington, DC, USA: IEEE Computer Society, 2007, pp. 1143–1148.
- [35] Y. A. Malkani and L. D. Dhomeja, “Secure device association for ad hoc and ubiquitous computing environments,” in *Proceedings of the International Conference on Emerging Technologies*, ser. ICET'09. IEEE, 2009, pp. 437–442.
- [36] A. Kumar, N. Saxena, G. Tsudik, and E. Usun, “A Comparative Study of Secure Device Pairing Methods,” *Pervasive and Mobile Computing*, vol. 5, no. 6, pp. 734–749, 2009.
- [37] M. K. Chong and H. Gellersen, “Usability classification for spontaneous device association,” *Personal and Ubiquitous Computing*, vol. 16, no. 1, pp. 77–89, 2012.
- [38] —, “How users associate wireless devices,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI'11. New York, NY, USA: ACM, 2011, pp. 1909–1918.
- [39] I. Ion, M. Langheinrich, P. Kumaraguru, and S. Capkun, “Influence of user perception, security needs, and social factors on device pairing method choices,” in *Proceedings of the 6th Symposium on Usable Privacy and Security*, ser. SOUPS'10. New York, NY, USA: ACM, 2010, pp. 6:1–6:13.
- [40] J. Canny, “The Future of Human-Computer Interaction,” *Queue-HCI*, vol. 4, no. 6, pp. 24–32, 2006.
- [41] S. Mitra and T. Acharya, “Gesture recognition: A survey,” *IEEE Trans. Systems, Man, and Cybernetics, Part C*, vol. 37, no. 3, pp. 311–324, 2007.
- [42] R. Dhamija, J. D. Tygar, and M. Hearst, “Why phishing works,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI'06. New York, NY, USA: ACM, 2006, pp. 581–590.
- [43] J. Suomalainen, J. Valkonene, and N. Asokan, “Standards for security associations in personal networks: a comparative analysis,” *International Journal of Security and Networks*, vol. 4, no. 1/2, pp. 87–100, 2009.

- [44] R. Kainda, I. Flechais, and A. W. Roscoe, "Usability and security of out-of-band channels in secure device pairing protocols," in *Proceedings of the 5th Symposium on Usable Privacy and Security*, ser. SOUPS'09. New York, NY, USA: ACM, 2009, pp. 11:1–11:12.
- [45] A. Kobsa, R. Sonawalla, G. Tsudik, E. Uzun, and Y. Wang, "Serial hook-ups: A comparative usability study of secure device pairing methods," in *Proceedings of the 5th Symposium on Usable Privacy and Security*, ser. SOUPS'09. New York, NY, USA: ACM, 2009, p. 12.
- [46] Bluetooth special interest group. Simple pairing whitepaper. [Online]. Available: <https://developer.bluetooth.org/DevelopmentResources/Pages/White-Papers.aspx>
- [47] R. Mayrhofer and H. Gellersen, "Shake well before use: Authentication based on accelerometer data," in *Proceedings of the 5th International Conference on Pervasive Computing*, ser. Pervasive'07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 144–161.
- [48] F. X. Lin, D. Ashbrook, and S. White, "RhythmLink: Securely pairing i/o constrained devices by tapping," in *Proceedings of the 24th Symposium on User Interface Software and Technology*, ser. UIST'11. New York, NY, USA: ACM, 2011, pp. 263–272.
- [49] M. Beigl, "Point and click - interaction in smart environments," in *Proceedings of the 1st International Symposium on Handheld and Ubiquitous Computing*, ser. HUC'99. New York, NY, USA: ACM, 1999, pp. 311–313.
- [50] J. Rekimoto, "Synctap: Synchronous user operation for spontaneous network connection," *Personal Ubiquitous Computing*, vol. 8, no. 2, pp. 126–134, 2004.
- [51] L. E. Holmquist, F. Mattern, B. Schiele, P. Alahuhta, M. Beigl, and H. W. Gellersen, "Smart-Its Friends: A technique for users to easily establish connections between smart artefacts," in *Proceedings of the 3rd International Conference on Ubiquitous Computing*, ser. UbiComp'01. London, UK: Springer-Verlag, 2001, pp. 116–122.
- [52] D. G. Park, J. K. Kim, S. J. Bong, J. H. Hwang, C. H. Hyung, and S. W. Kang, "Tap: Touch-and-play," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI'06. New York, NY, USA: ACM, 2006, pp. 677–680.

- [53] C. Peng, G. Shen, Y. Zxang, and S. Lu, "Point and connect: Intention-based device pairing for mobile phone users," in *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys'09. New York, NY, USA: ACM, 2009, pp. 137–150.
- [54] A. D. Wilson and R. Sarin, "Bluetable: Connecting wireless mobile devices on interactive surfaces using vision-based handshaking," in *Proceedings of the International Conference on Graphics Interface*, ser. GI'07. New York, NY, USA: ACM, 2007, pp. 119–125.
- [55] A. Studer, T. Passaro, and L. Bauer, "Don't bump, shake on it: the exploitation of a popular accelerometer-based smart phone exchange and its secure replacement," in *Proceedings of the 27th Annual Conference on Computer Security Applications*, ser. ACSAC'11. New York, NY, USA: ACM, 2011, pp. 333–342.
- [56] E. Jones, J. Alexander, A. Andreou, P. Irani, and S. Subramanian, "Gestext: Accelerometer-based gestural entry systems," in *Proceedings of the 28th Conference on Human Factors in Computing Systems*, ser. CHI'10. New York, NY, USA: ACM, 2010, pp. 2173–2182.
- [57] E. Choi, W. Bang, C. S., T. K. D. Yang, and S. Kim, "Beatbox music phone: Gesture-based interactive mobile phone using tri-axis accelerometer," in *Proceedings of the IEEE International Conference on Industrial Technology*, ser. ICIT'05. IEEE, 2005, pp. 97–102.
- [58] S. Agrawal, I. Constandache, and S. Gaonkar, "Phonepoint pen: Using mobile phones to write in air," in *Proceedings of the 1st international workshop on Networking, Systems, and Applications for Mobile Handhelds*, ser. MobiHeld'09. New York, NY, USA: ACM, 2009, pp. 1–6.
- [59] S. N. Patel, J. S. Pierce, and G. D. Abowd, "A gesture-based authentication scheme for untrusted public terminals," in *Proceedings of the 17th Symposium on User Interface Software and Technology*, ser. UIST'04. New York, NY, USA: ACM, 2004, pp. 157–160.

- [60] M. K. Chong and G. Marsden, “Exploring the use of discrete gestures for authentication,” in *Proceedings of the 12th IFIC TC13 Conference on Human Computer Interaction*, ser. INTERACT’09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 97–102.
- [61] V. M. Mantyla, “Discrete hidden markov models with application to isolated user dependent hand gesture recognition,” VTT Publications, Tech. Rep., 2001.
- [62] J. Mantyjarvi, J. Kela, P. Korpipaa, and S. Kallio, “Enabling fast and effortless customization in accelerometer based gesture interaction,” in *Proceedings of the 3rd International Conference on Mobile and Ubiquitous Multimedia*, ser. MUM’04. New York, NY, USA: ACM, 2004, pp. 25–31.
- [63] T. Schlomer, B. Poppinga, N. Henze, and S. Boll, “Gesture recognition with a wii controller,” in *Proceedings of the International Conference on Tangible and Embedded Interaction*, ser. TEI’08. New York, NY, USA: ACM, 2008, pp. 11–14.
- [64] S. J. Cho, E. Choi, W. C. Bang, J. Yang, J. Sohn, D. Y. Kim, Y. B. Lee, and S. Kim, “Two-stage recognition of raw acceleration signals for 3d-gesture-understanding cell phones,” in *Proceedings of the 10th International Workshop on Frontiers in Handwriting Recognition*, ser. IWFHR’06, 2006, p. 6.
- [65] J. Wu, G. Pan, D. Zhang, G. Qi, and S. Li, “Gesture recognition with a 3d accelerometer,” in *Proceedings of the 6th International Conference on Ubiquitous Intelligence and Computing*, ser. UIS’09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 25–38.
- [66] G. Niezen and G. P. Hancke, “Gesture recognition as ubiquitous input for mobile phones,” in *Proceedings of the International Workshop on Devices that Alter Perception*, ser. DAP’08. New York, NY, USA: ACM, 2008, p. 5.
- [67] J. Liu, Z. Wang, L. Zhong, J. Wickramasuriya, and V. Vasudevan, “uWave: Accelerometer-based gesture recognition and its applications,” in *Proceedings of the 7th IEEE International Conference on Pervasive Computing and Communications*, ser. PerCom’09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 1–9.
- [68] D. H. Wilson and A. Wilson, “Gessture recognition using the xwand,” CMU Robotics Institute, Tech. Rep., 2004.

- [69] G. Niezen, “The optimization of gesture recognition techniques for resource-constrained devices,” Master’s thesis, University of Pretoria, South Africa, 2009.
- [70] M. Muller, *Information Retrieval for Music and Motion*. Springer, 2007, ch. Dynamic Time Wrapping.
- [71] C. Myers and L. R. Rabiner, “A comparative study of several dynamic time wrapping algorithms for connected word recognition,” *The Bell System Technical Journal*, vol. 60, no. 7, pp. 1389–1409, 1981.
- [72] (July, 2014) SQLite Database Library. [Online]. Available: <http://www.sqlite.org/>
- [73] A. Ayman, “Interaction with gestures in ubiquitous environments,” Ph.D. dissertation, University of Tsukuba, Japan, 2010.
- [74] S. Creese, M. Goldsmith, B. Roscoe, and M. Xiao, “Bootstrapping multi-party ad-hoc security,” in *Proceedings of the ACM Symposium on Applied Computing*, ser. SAC’06. New York, NY, USA: ACM, 2006, pp. 369–375.
- [75] S. Laur and S. Pasini, “SAS-based group authentication and key agreement protocols,” in *Proceedings of the 11th International Conference on Practice and Theory in Public Key Cryptography*, ser. PKC’08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 197–213.
- [76] L. H. Nguyen and A. W. Roscoe, “Authenticating ad-hoc networks by comparison of short digests,” *Information and Computation*, vol. 206, no. 2-4, pp. 250–271, 2008.
- [77] Y. Lin, A. Studer, H. Hsiao, J. McCune, K. Wang, M. Krohn, P. Lin, A. Perrig, H. Sun, and B. Yang, “SPATE: small-group PKI-less authenticated trust establishment,” in *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys’09. New York, NY, USA: ACM, 2009, pp. 1–14.
- [78] C. O. Chen, C. Chen, C. Kuo, Y. Lai, J. M. McCune, A. Studer, A. Perrig, B. Yang, and T. Wu, “GANGS: Gather, authenticate and group securely,” in *Proceedings of the 14th International Conference on Mobile Computing and Networking*, ser. MobiCom’08. New York, NY, USA: ACM, 2008, pp. 92–103.

- [79] R. Kainda, I. Flechais, and A. W. Roscoe, “Two heads are better than one: Security and usability of device associations in group scenarios,” in *Proceedings of the 6th Symposium on Usable Privacy and Security*, ser. SOUPS’10. New York, NY, USA: ACM, 2010, pp. 5:1–5:13.
- [80] R. Nithyanand, N. Saxena, G. Tsudik, and E. Uzun, “Groupthink: usability of secure group association for wireless devices,” in *Proceedings of the 12th International Conference on Ubiquitous Computing*, ser. UbiComp’10. New York, NY, USA: ACM, 2010, pp. 331–340.
- [81] L. Ming, S. Yu, J. D. Guttman, W. Lou, and R. Ren, “Secure ad-hoc trust initialization and key management in wireless body area network,” *ACM Transactions on Server Networks*, vol. 9, no. 2, pp. 18:1–18:35, 2013.
- [82] S. L. Keoh, E. Lupu, and M. Sloman, “Securing body sensor networks: Sensor association and key management,” in *Proceedings of the IEEE International Conference on Pervasive Computing and Communications*, ser. PerCom’09. IEEE, 2009, pp. 1–6.
- [83] M. Li, S. Yu, W. Lou, and K. Ren, “Group device pairing based secure sensor association and key management for body area networks,” in *Proceedings of the 29th International Conference on Information Communications*, ser. InfoCom’10. Piscataway, NJ, USA: IEEE, 2010, pp. 2651–2659.
- [84] A. Lucero, T. Jokela, A. Palin, V. Aaltonen, and J. Nikara, “EasyGroups: binding mobile devices for collaborative interactions,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI’12. New York, NY, USA: ACM, 2012, pp. 2189–2194.
- [85] M. K. Chong, F. Kawsar, and H. Gellersen, “Spatial co-location for device association : The connected object way,” in *Proceedings of the International Workshop on Networking and Object Memories for the Internet of Things*, ser. NoME-IoT’11. New York, NY, USA: ACM, 2011, pp. 21–26.
- [86] E. Uzun, N. Saxena, and A. Kumar, “Pairing devices for social interactions: A comparative usability evaluation,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI’11. New York, NY, USA: ACM, 2011, pp. 2315–2324.

- [87] C. Kuo, A. Studer, and A. Perrig, “Mind your manners: Socially appropriate wireless key establishment for groups,” in *Proceedings of the International Conference on Wireless Network Security*, ser. WuSec’08. New York, NY, USA: ACM, 2008, pp. 125–130.
- [88] M. K. Chong and H. Gellersen, “How groups of users associate wireless devices,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI’13. New York, NY, USA: ACM, 2013, pp. 1559–1568.
- [89] J. P. Hubaux, B. Levente, and S. Capkun, “The quest for security in mobile ad-hoc networks,” in *Proceedings of the 2nd International Symposium on Mobile Ad-hoc Networking and Computing*, ser. MobiHoc’01. New York, NY, USA: ACM, 2001, pp. 146–155.
- [90] C. E. Epp, “Relationship management: Secure collaboration in a ubiquitous environment,” *Pervasive Computing, IEEE*, vol. 2, no. 2, pp. 62–71, 2003.
- [91] O. Chagnadorj and J. Tanaka, “Gesture input as out-of-band channel,” *Journal of Information Processing Systems*, vol. 10, no. 1, pp. 92–102, 2014.
- [92] (July, 2014) The Legion of the Bouncy Castle. [Online]. Available: <http://www.bouncycastle.org>
- [93] (July, 2014) Zxing Library. [Online]. Available: <https://github.com/zxing/zxing>
- [94] M. Steiner, G. Tsudik, and M. Waidner, “Key agreement in dynamic peer groups,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 11, no. 8, pp. 769–780, 2000.
- [95] W. G. Tzeng and S. J. Tzeng, “Round-efficient conference key agreement protocols with provable security,” in *Proceedings of the 6th International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology*, ser. ASIACRYPT’00. Berlin, Heidelberg: Springer-Verlag, 2000, pp. 614–628.
- [96] Y. Kim, A. Perrig, and G. Tsudik, “Simple and fault-tolerant key agreement for dynamic collaborative groups,” in *Proceedings of the 7th International Conference on Computer and Communications Security*, ser. CCS’00. New York, NY, USA: ACM, 2000, pp. 235–244.

-
- [97] B. Lavelle, D. Byrne, G. Jones, and A. F. Smeaton, “Bluetooth friendly names: Bringing classic HCI questions into the mobile space,” in *Proceedings of the 21th British HCI Group Annual Conference*. British Computer Society, 2007, p. 4.
- [98] A. Liu and P. Ning, “TinyECC: A configurable library for elliptic curve cryptography in wireless sensor networks,” in *Proceedings of the 7th International Conference on Informatin Processing in Sensor Networks*, ser. ISPN’08. Washington, DC, USA: IEEE Computer Society, 2008, pp. 245–256.

List of Publications

Journals

Oyuntungalag Chagnaadorj and Jiro Tanaka. "Gesture Input as an Out-of-band Channel," *Journal of Information Processing Systems*, Vol.10, No.1, 2014, pp.92-102.

Conference proceedings

Oyuntungalag Chagnaadorj and Jiro Tanaka. "MimicGesture: Secure Device Pairing with Accelerometer-based Gesture Input," *Proceedings of the 7th International Conference on Ubiquitous Information, Technologies and Applications (CUTE 2012)*, *LNEE 214*, pp.59-67, Hong Kong, China, December 20-22, 2012.

Oyuntungalag Chagnaadorj and Jiro Tanaka. "MobileCA: Accumulative Secure Group Association with a Certification Path," *Proceedings of the 4th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN 2013)*, *Procedia Computer Science 21*, pp.242-249, Niagara Falls, Ontario, Canada, October 21-24, 2013