

## SPECIAL ISSUE PAPER

# A fingerprint based bio-cryptographic security protocol designed for client/server authentication in mobile computing environment

Kai Xi, Tohari Ahmad, Fengling Han and Jiankun Hu\*

School of Computer Science and Information Technology, RMIT University, Melbourne 3001, Australia

## ABSTRACT

With fast evolution of mobile devices and mobile network, the need of protecting user sensitive information locally and performing secure user authentication remotely become evermore increasing. Bio-cryptography is emerging as a powerful solution which can combine the advantages of conventional cryptography and biometric security. In this paper, we present an efficient bio-cryptographic security protocol designed for client/server authentication in current mobile computing environment, with a reasonable assumption that server is secure. In this protocol, fingerprint biometric is used in user verification, protected by a computationally efficient Public Key Infrastructure (PKI) scheme, Elliptic Curve Cryptography (ECC). The genuine fingerprint information is hidden in the feature vault which is the mixture of genuine and chaff features. Fingerprint features are not only used for biometric verification but also for cryptographic key generation. Our security analysis shows that the proposed protocol can provide a secure and trustworthy authentication of remote mobile users over insecure network. Experimental results on public domain database show an acceptable verification performance. We also tested the computational costs and efficiency of our protocol on the CLDC emulator using Java ME (previous J2ME) programming technology. The simulation results prove that the proposed protocol suits current mobile environment. Copyright © 2010 John Wiley & Sons, Ltd.

## KEYWORDS

bio-cryptography, mobile computing, biometric, fingerprint authentication

### \*Correspondence

Jiankun Hu, School of Computer Science and Information Technology, RMIT University, Melbourne 3001, Australia.

E-mail: jiankun@cs.rmit.edu.au

## 1. INTRODUCTION

Mobile devices such as mobile phones have evolved from simple voice communication dominate electronic devices to powerful digital handsets with multiple roles such as digital camera, video recorder, radio, MP3 player, web browser, gaming terminal, GPS navigator, and mobile TV. Recently, more and more data-centric services are provided over mobile networks. Mobile devices will store much more information such as personal data and financial information than previous handsets. Mobile users could use the services such as trading stocks, processing micro-payment and managing bank accounts, and using online data storage services.

The portability of mobile devices and the convenience of mobile services make them increasingly attractive. However, they are also exposed to the risk of information leakage and illegitimate use. When mobile devices are lost or stolen, not only the devices themselves but also the stored

information may fall into the wrong hands. Such sensitive information can be a stored PIN or a cryptographic key belonging to a client account of an e-commerce system. Attackers may utilize the obtained handheld devices and stored information to cheat the remote server for authorization. It is widely recognized that current mobile computing environment requires higher levels of protection. In Reference [1], the authors organized a survey among 230 business professionals. It is reported that 81% were utilizing their handsets to store valuable data. Seventy per cent expected their handsets to possess security mechanism. Sixty nine per cent are pleased to pay extra for such security module while 1% only prefers free security service. Security has become a major concern in the Client-Server architecture [31,33,34,36,37].

Cryptography is a conventional method of authenticating users and protecting communication messages in insecure networks. Only the user who possesses the correct cryptographic key can access the encrypted content.

Cryptographic algorithms can be coarsely grouped into two types: symmetric key cryptography and public key cryptography. The symmetric key cryptography is fast which makes it suitable for encrypting long message. However, exchanging cryptographic keys between a sender and a receiver is a challenge. A public key cryptography is more promising since it does not require a secure exchange of session keys as when using symmetric key algorithms. Nevertheless, in practice, the slow computational speed restricts PKI to be applied in short message encryption/decryption only. The most widely used public key cryptographic algorithm is the RSA algorithm proposed in Reference [2]. Since the low processing efficiency, RSA is normally employed to encrypt short symmetric key. In 1985, Miller [3] and Koblitz [4] proposed a public key cryptographic scheme called elliptic curve cryptography (ECC). The ECC has a smaller key size which offers the same security strength as the RSA. This will make it more preferable for mobile devices where small memories and low computational powers are expected [29].

However, cryptography has its own drawbacks. For instance, an attacker may obtain the cryptographic key *via* illegal ways and then pretend to be an authentic user. It is difficult to detect and prevent this type of attack at the server side because most recent security system identifies users based on the (1) knowledge: something you know such as passwords, personal identification numbers (PIN) or a cryptographic key, (2) token: something you have such as Subscriber Identity Module (SIM) cards. Passwords, PIN and key can be guessed out while a SIM card is usually left in the handset which may be obtained by illegal users. Servers can only verify whether the knowledge and tokens provided from the remote client side are authentic or not. Nevertheless, it is extremely hard to identify who provides such knowledge or tokens [30,35,39,40].

Biometric techniques offer a natural and reliable solution for identifying individuals. Biometric based security systems recognize a person by physiological characteristics like fingerprint, face, iris, palm, etc. or behavioral characteristics like signature, voice, gait, keystroke dynamics, etc. In principal, biometrics can validate genuine user's presence, hence enhancing the authentication reliability. Biometric traits offer three main benefits: (1) universality—every person possesses the biometric features, (2) uniqueness—it is unique from person to person, (3) performance stability—its properties remain stable during one's lifetime. These characteristics enable biometric-based authentication and identification systems to provide higher level protection than conventional knowledge based and token based system [5].

Biometric and cryptography could become complementary to each other. It is reasonable and feasible to incorporate biometric into the cryptographic infrastructure. Soutar *et al.* [6—8] proposed a key-binding algorithm using correlation-based fingerprint matching method. In the algorithm, a cryptographic key and the corresponding user's fingerprint image are bound at the enrollment stage. Key retrieval process is protected by fingerprint verification. Correct keys can only be released upon successful

authentication. If the biometric authentication fails, an 'authentication failed' message will be returned. However, the downside of this scheme is obvious. The biometric verification and cryptographic component are decoupled which result in that cryptographic key can be achieved easily if attackers bypass the biometric security module. In addition, their work is based on the unrealistic condition that the query fingerprint impression and template are perfectly aligned. No performance evaluation was reported in literature.

Recent significant research outcome on bio-cryptographic includes biohashing, cancelable template, fuzzy extractor and fuzzy vault. For biohashing technique, Teoh *et al.* propose the Random Multispace Quantization (RMQ) method in Reference [9]. In the method, discriminative projections of the face template are extracted by PCA or FDA. Afterward, the obtained biometric feature vector is mapped onto a sequence of random subspaces. Savvides and Vijayakumar proposed the cancelable face filter method in Reference [10] where the authors convolve the training image with random convolution kernel. Different templates can be obtained from the same biometric by varying the convolution kernels thus enabling the cancelability of the templates. Other cancelable biometric can be found in References [11] and [12]. Fuzzy extractor [13] is a type of key generating approach designed to convert noisy data, e.g. biometric features, into cryptographic keys. It is a combination of a primitive called a Secure Sketch and a Strong Randomness Extractor. The Secure Sketch generates public help data which are related to the input but does not reveal biometric information. The Randomness Extractor is used to map the non-uniform input to a uniformly distributed string, in order to achieve the maximum information entropy. Juels and Sudan [14] proposed a cryptographic construction called fuzzy vault construct. In Reference [15], the authors presented its application for fingerprint-based security system, called fingerprint fuzzy vault. The general idea is to hide the cryptographic key in a scrambled list which is composed of genuine fingerprint features and fabricated chaff features. The security strength of the fuzzy vault is based on the infeasibility of the polynomial reconstruction problem. The variation of fingerprint fuzzy vault can be found in Reference [16]. Although there exist many bio-cryptographic methods, most of these solutions cannot be easily implemented on mobile devices. Also how to combine the advantages of biometrics and the conventional PKI framework has not been explored. This will be the focus of this paper.

In this paper, a new bio-cryptographic protocol is proposed. In the authentication process of the protocol, the fingerprint biometric is employed to work with the ECC public key cryptography. The basic idea is to transfer the locally matched fuzzy vault index to the central server for biometric authentication using the PKI which offloads the computation demand to the central server. Biometric cancelable keys are generated while minutia details are never exposed externally. Establishment of symmetric session keys does not need a conventional key exchange process

which further reduces the vulnerability risk. The accuracy of the proposed protocol is evaluated using the public domain fingerprint database NIST 24. Furthermore, we implement our algorithm on Java ME emulator for the test of memory usage and computational efficiency. Note that we assume that the central server is secure and computing resource rich. This assumption is practical due to following reasons: (i) a central server is often physically secure. This will significantly reduce the level of security challenge compared with the scenario such as mobile device where physical access to the attacker is allowed, (ii) in many PKI systems, a trust party is also needed or assumed.

The rest of the paper is organized as follows. Section 2 introduces the proposed bio-cryptographic protocol. The related security analysis is presented in Section 3. Section 4 provides experimental results and the related issues of the Java ME implementation. Our conclusions and future work are in Section 5.

## 2. PROPOSED BIO-CRYPTOGRAPHIC SECURITY PROTOCOL

Most existing biometric-cryptography systems are based on the idea of cryptographic key release where the biometric authentication is decoupled from the cryptographic part. The biometric module can only output either an 'accept' or 'reject' decision. If accept, the cryptographic key will be released from the key management module. If reject, the system will not release any key. Here, biometric module provides a wrapper mechanism for the cryptographic module which makes the whole system vulnerable to attacks like the Trojan horse. A Trojan horse program may tamper with the biometric authentication module and simply inject an 'accept' command to the key management subsystem.

In our protocol, biometrics and cryptography are seamlessly integrated. The proposed bio-cryptographic protocol consists of two phases namely user registration and user authentication, which are presented in the following subsections.

### 2.1. User registration

In the user registration phase, a user  $U$  registers himself/herself into the system. The process should be attack free. It is a secure and feasible way that a new user is required to present at the server side to accomplish this process. User biometric template generation should be under the control of professionals, such as system administrators, in order to guarantee the high authentication accuracy.

To register, at first system will randomly generate a unique ID  $ID_u$  for a user  $U$ . For convenience we assume the length of  $ID_u$  be 64 bits (8 bytes). Then  $U$  imprints multiple fingerprint impressions  $fg_i$  ( $i = 1, 2, \dots, N$ ) where  $N$  is the total number of acquired impressions. Here the location  $(x, y)$  and orientation angle  $\theta$  are used to represent one minutia.  $(x, y)$  is a Cartesian coordinate and  $\theta \in (0, 2\pi)$ . A

minutia can be described as  $m = (x, y, \theta)$ . Each impression  $fg_i$  generates a feature set  $M_i = \{m_1, m_2, \dots, m_n\}$ , where  $n$  is the total number of minutiae in a fingerprint impression.

It should be noted that normally the input fingerprint impressions are not aligned. Shift and rotation usually exist among different impressions. Even the same minutia obtained from different impressions could show different  $(x, y)$  and  $\theta$ . Alignment processing is necessary. Often, a reference point (usually the singular point) is required for alignment. Errors in locating the singular point could bring about false rejections. Commercial embedded fingerprint recognition products such as UPEK TCS5B [17] and AuthenTec AES 2510 [18] possess the capability of extracting the minutiae as well as singular point. These products are inexpensive, for example TCS5B only cost 8 US dollars [19]. The corresponding solutions have been successfully adopted by Toshiba, ASUS, Lenovo, mainly on laptops and PDAs. Because minutiae retrieval and singular point detection are not part of our intended contribution, a commercial software Verifinger 5.0 [20] for minutiae extraction and the FOMFE [21,32,38] approach for singular point detection were chosen for our experiment due to our available experience. Other commercial products can also be used. The singular point acquired from each impression is  $sp_i$ , described using  $(x, y, \theta)$ . Each  $M_i$  associates with one  $sp_i$ . If no singular point is detected, the system will ask the user either to retry or to change a finger.

For each  $\{M_i, sp_i\}$ , the system will create a new coordinate system whose origin is  $sp_i$ . All minutiae points of  $M_i$  are transformed based on the new coordinate system. The new obtained minutia point sets are  $Q_i$ , ( $i = 1, 2, \dots, N$ ).

Multiple impressions are used to improve the authentication accuracy. We conduct a cross-matching on  $Q_i$  to find the matched and unmatched minutiae. All minutiae will be given a unique index number  $SN$  and included in a lookup table  $T$ . The matched minutiae pair will be taken as one minutia. All minutiae should be fully separated. Two minutiae are considered separated if their distance  $d$  or orientation angle difference  $\Delta\theta$  is greater than pre-defined thresholds  $d_{\text{threshold}}$  and/or  $\theta_{\text{threshold}}$ .

The distance between any two minutiae  $m_i = (x_i, y_i, \theta_i)$  and  $m_j = (x_j, y_j, \theta_j)$  is defined as

$$d(m_i, m_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \quad (1)$$

The angle difference of  $m_i = (x_i, y_i, \theta_i)$  and  $m_j = (x_j, y_j, \theta_j)$  is

$$\Delta\theta(m_i, m_j) = \min(|\theta_i - \theta_j|, (2\pi - |\theta_i - \theta_j|)) \quad (2)$$

In order to increase the security strength, a great number of chaff points are generated and inserted into  $T$  with a given index number. A chaff point  $c = (x, y, \theta)$  is randomly generated under the condition that  $d$  or  $\Delta\theta$  between  $c$  and any genuine minutiae are greater than  $d_{\text{threshold}}$  and  $\theta_{\text{threshold}}$ , respectively. The central server stores another list

$L$  which records the index numbers in  $T$  that correspond to all genuine minutiae.

At the client side, the mobile device also stores the lookup table  $T$  and  $ID_u$  for the future authentication. However, it should be noted that the mobile client does not possess the list  $L$ . Therefore, without the help of  $L$ , it is infeasible to extract sufficient genuine biometric data, such as nine genuine minutiae, directly from the lookup table  $T$ .

## 2.2. User authentication

In the user authentication phase, a user  $U'$  imprints multiple fingerprint impressions  $fp'_i$  ( $i = 1, 2, \dots, N$ ) using the scanner equipped on the device where  $N$  is the total number of acquired impressions. The biometric module on the mobile device extracts the minutiae points and the singular point  $\{M'_i, sp'_i\}$  from  $fp'_i$ . To perform an alignment, a new coordinate system, whose origin is  $sp'_i$ , is established. All minutiae points of  $M'_i$  are transformed based on the new coordinate system. The new obtained minutia point sets are  $Q'_i$ , ( $i = 1, 2, \dots, N$ ). A cross matching is performed between each  $Q'_i$  pair in order to find reliable minutiae and remove spurious minutiae. It is reasonable that most reliable and stable minutiae points will appear multiple times in different impressions. After cross matching, new minutiae set  $Qs'$  is acquired where minutiae points in  $Qs'$  are the points that appear in at least two impressions. If the number of minutiae in  $Qs'$  is less than the pre-defined threshold  $n_{\text{threshold}}$ , it is considered as failure to capture and no further processing will be executed. All minutiae of  $Qs'$  are compared with the points of lookup table  $T$ . If there is a match, the index number of matched point in  $Qs'$  will be added to a list  $L_0$ . However, in our protocol the length of  $L_0$  being sent to the central server must be capped. This is because too many minutiae selected from the fuzzy vault list  $T$  participating in the authentication will increase the false acceptance rate. It is often that the length of  $L_0$  is larger than the cap value. Therefore, we must reduce this list to a shorter list  $L'$ . The length of  $L'$  can be determined experimentally by the demands on false acceptance and false rejection rate.

To prevent information leakage,  $U'$  only transmits the index numbers instead of matched minutiae information to the central server for an authentication. The process can be described as:  $U'$  obtains the server's certificate from CA and encrypts  $L'$ ,  $ID_{u'}$  and a timestamp  $Ts'$  using the server's public key  $K_{sr}^{pb}$ .  $C_1$  is the output ciphertext, where

$$C_1 = E_{K_{sr}^{pb}}(ID_{u'}, L', Ts') \quad (3)$$

$C_1$  is then transmitted to the central server  $Sr_{cen}$ .

When receives  $C_1$ ,  $Sr_{cen}$  decrypt  $C_1$  using its private key  $K_{sr}^{pri}$ .  $L'$  and  $ID_{u'}$  is obtained as

$$(ID_{u'}, L', Ts') = D_{K_{sr}^{pri}}(C_1) \quad (4)$$

$ID_{u'}$  is used to find the profile of the claimed user  $ID_u$ .  $Ts'$  is used to check if it is a replay attack. If  $ID_{u'}$  can be

found on  $Sr_{cen}$ ,  $L'$  will be compared with  $L$  that belongs to  $ID_u$ . Suppose  $L'$  and  $L$  has  $n$  matched index numbers. Authentication is successful if  $n$  is greater than  $n_{\text{threshold}}$  and vice versa.

After a successful authentication,  $Sr_{cen}$  return the encrypted version of  $L'$ ,  $ID_{u'}$  and  $Ts'$  to  $U'$ . They are encrypted by the public key  $K_u^{pb}$  of  $U$  obtained from CA. Encryption process can be expressed as

$$C_2 = E_{K_u^{pb}}(ID_{u'}L', Ts') \quad (5)$$

The ciphertext  $C_2$  is then transmitted to  $U'$ , as not only a acknowledgement but also an key agreement.

$U'$  is able to decrypt  $C_2$  successfully using his own private key  $K_u^{pri}$  only if  $K_u^{pri} = K_u^{pri}$ . The process that extracts  $L'$ ,  $ID_{u'}$  and  $Ts'$  from  $C_2$  can be described as

$$(ID_{u'}L', Ts') = D_{K_u^{pri}}(C_2) \quad (6)$$

Here,  $U'$  verifies whether the received  $L'$ ,  $ID_{u'}$  and  $Ts'$  are exact the same as the original ones transmitted to  $Sr_{cen}$  before. The purpose is the mutual authentication so that  $U'$  is able to verify if  $C_2$  comes from the genuine server  $Sr_{cen}$  which he/she intends to connect to.

If mutual authentication successful, both  $U'$  and  $Sr_{cen}$  will generate a biometric-based session key  $K_{bio}$  (symmetric cryptographic key) using the minutiae whose index numbers present in  $L'$ . Further communication between the mobile client  $U'$  and the central server  $Sr_{cen}$  will be protected by  $K_{bio}$ .

Figure 1 demonstrates the working flow of a successful user authentication process.

## 3. SECURITY STRENGTH

Attacks have posed a grave threat to the current security system especially in mobile environment. For a bio-cryptosystem, not only the cryptographic information, e.g. the key, but also the biometric data should be protected against attacks. In this section, we list several typical attacks and analyze how the proposed protocol defends against them. In reality, most servers have been assumed and treated as 'secure' even though they are not perfectly secure. Sensitive information is usually stored on the server e.g. personal information, banking details, password, etc. Most popular B2C, B2B, ERP systems are using client-server architecture, all on the basis of that the server is secure. Therefore, this is a reasonable assumption. In mobile computing architecture, the major security weak points are mobile devices and the transmission channel. Compared with them, servers can offer relatively higher security strength. Server-side attack is not our research focus and beyond the scope of this paper.

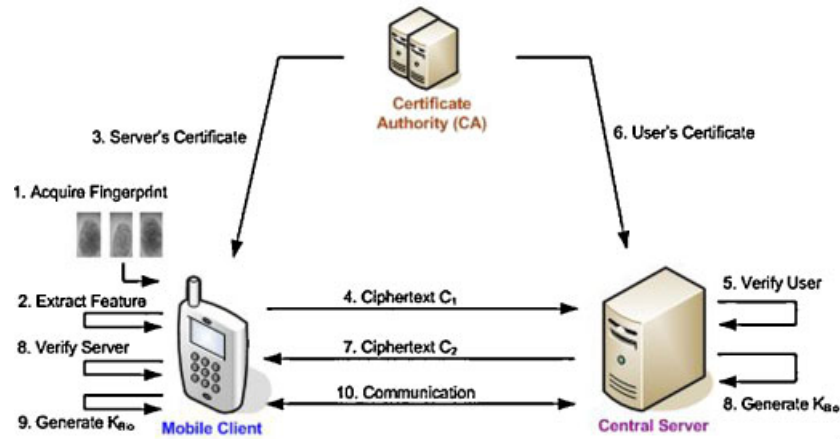


Figure 1. Process of a successful user authentication.

### 3.1. Trojan horse attack

In such type of attacks, attackers may use a Trojan horse program to replace the system module and bypass its security mechanism. Conventional biometric key release systems store a static key in the hardware that can be illegally captured by attackers *via* overriding the biometric authentication module. Our protocol overcomes this flaw from following three points:

- (1) Even in the worst case that an attacker compromise the private key of the PKI which has been stored in the mobile device, it does not help break the system as the attacker has to pass the biometric authentication at the central server in the first place.
- (2) In our protocol, the session key  $K_{bio}$  is generated directly from biometric feature rather than pseudo-random number generator. Hence,  $K_{bio}$  is unpredictable. Unless attackers obtain the genuine biometric data, otherwise they are not easy to guess out  $K_{bio}$ .
- (3) We do not store any static symmetric key at either mobile client side or server side.  $K_{bio}$  is dynamically generated which means each time a mobile user communicates with the server  $K_{bio}$  tends to be different.

In our protocol, the biometric and cryptography are seamlessly integrated. Cryptographic key cannot be independently generated without biometric information. Hence, it is robust against Trojan horse attacks.

### 3.2. Brute force attack

In cryptanalysis, a brute force attack is a method of breaking a cryptographic scheme by systematically trying a great number of possibilities. The security strength of an algorithm cannot exceed its key length. In our protocol,  $K_{bio}$  is generated from a set of minutia points. Three attributes  $x$ ,  $y$ , and  $\theta$  of a minutia are used for the design of  $K_{bio}$ .

Since the mobile environment is quite open, attackers may capture the  $K_{bio}$ . There exists a risk that attackers trace  $K_{bio}$  back to the original biometric data. To prevent it, we use a non-invertible (cancelable) transformed version of biometric data to generate the  $K_{bio}$ . By the sake of computational efficiency we perform a simple transformation

$$R_{xy} = \text{Hash}(x|y) \bmod p \quad (2048 < p < 4096) \quad (7)$$

where  $\text{Hash}(x|y)$ , a one-way bijective hash function, is used to increase the randomness. In our implementation, a widely used cryptographic hash function SHA-224 [22] is employed. SHA-224 can generate a 224 bits hash value from an input message. The modulo operation is a non-injective function that can map different inputs to a same  $R_{xy}$ .  $p$  is a parameter that can be revoked. In real application, we could use time-stamp to generate  $p$ . By using different  $p$ , the output keys are uncorrelated with others. Both server and client do not require any pre-defined knowledge. For instance, when mobile client transmits an authentication request to the server, the moment  $t$  of sending out the message can be recorded and attached on the request message. After successfully verifying the client, the server sends an offset  $\Delta t$  back to client. Afterward, both server and client use the time  $t + \Delta t$  to obtain  $p$  and then generate the bio-key  $K_{bio}$ .

After performing Equation (7), the distribution of points become more random, as demonstrated in Figure 2.

Generating  $K_{bio}$  from  $R_{xy}$  is more secure than using  $x$ ,  $y$  directly. If attackers illegally obtain  $K_{bio}$ , it is difficult for them to retrieve the original minutiae information from  $K_{bio}$  because the process that transform  $x$ ,  $y$  to  $R_{xy}$  is non-invertible. In addition, when the system senses an attack,  $p$  will be revoked and reset so that a new group of  $K_{bio}$  can be generated.

Figure 2 illustrates the procedure of convert the raw biometric feature  $x$ ,  $y$  to  $R_{xy}$  *via* hash and modulo operations. The bar chart demonstrates the number of generated  $R_{xy}$  falling into different value ranges. It can be observed that

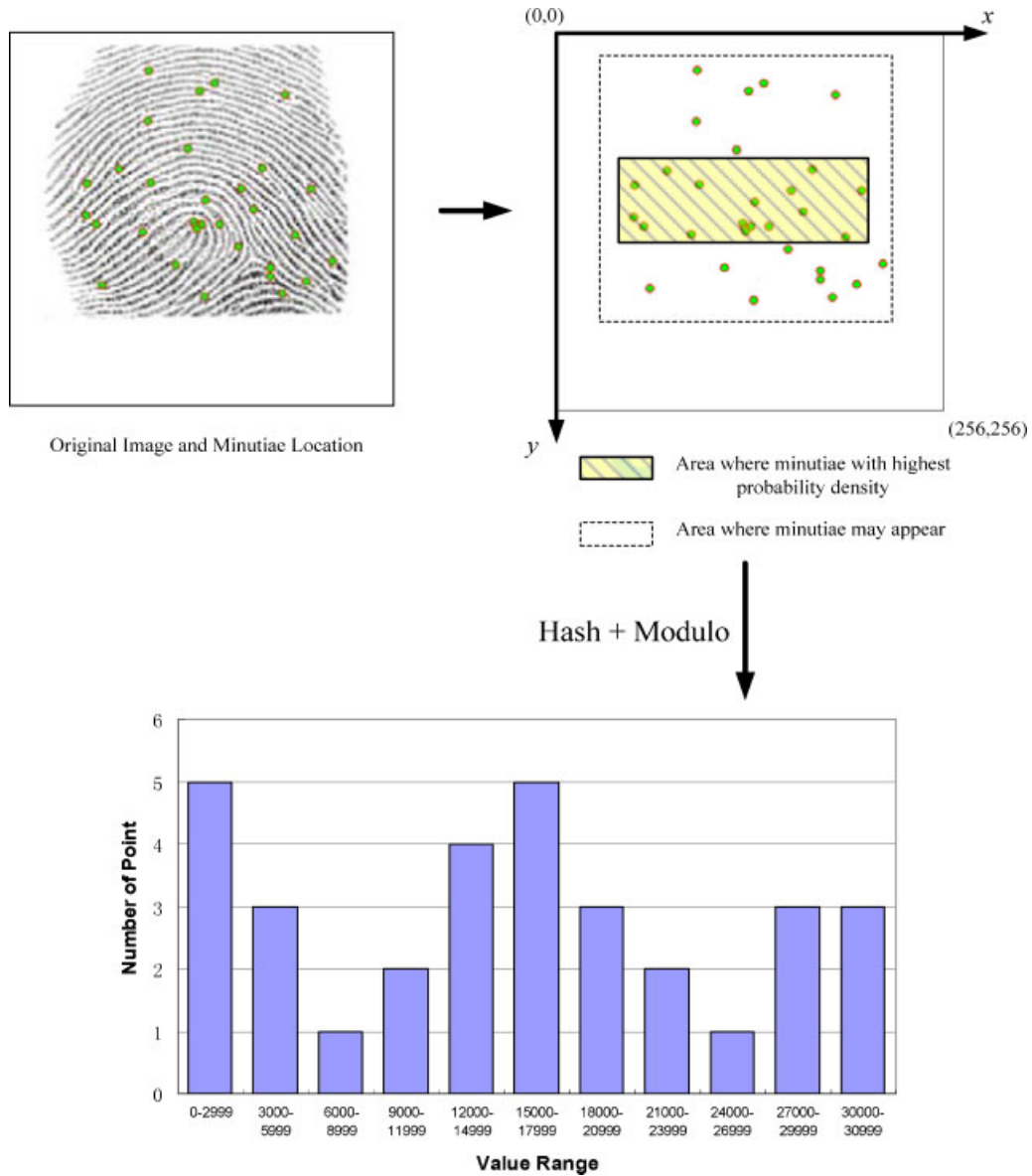


Figure 2. Perform hash function and modulo operation on minutiae coordinates.

the value of generated  $R_{xy}$  trend to be evenly distributed. The location relations between each minutiae pair in the original image have been eliminated during Hash-Modulo operation.

$R_{xy}$  and  $\theta$  are expressed using 12-bit and 9-bit stream, respectively. This can be simply achieved by converting the decimal values of  $R_{xy}$  and  $\theta$  to the binary representations. For instance, a minutia (57, 164, 132°) can be converted to

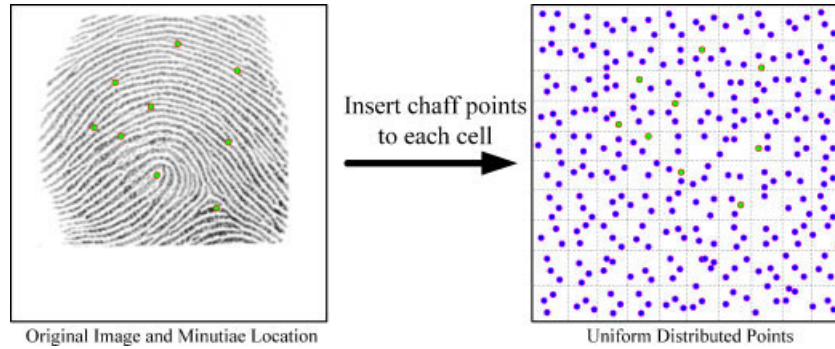
$$\begin{aligned}
 &(\text{Hash}(057164) \bmod 3849, 132) = (2846, 132) \\
 &= (101100011110010000100) \tag{8}
 \end{aligned}$$

One minutia can contribute 21 bits to  $K_{\text{bio}}$ . In our protocol, the number of selected minutiae for key generation is

10–16. Both genuine minutiae and chaff points can be used to generate  $K_{\text{bio}}$ . The key strength of  $K_{\text{bio}}$  originating from biometric data is 210–336 bits. Plus, the string  $\text{ID}_{\text{usr}}$  contributes additional 64 bits (8 bytes). Total key length of  $K_{\text{bio}}$  is 274–400 bits, which is considered to be secure enough for popular symmetric schemes such as the AES scheme against brute force attack.

### 3.3. Biometric template attack

In our protocol,  $K_{\text{bio}}$  originates from fingerprint data, i.e. minutiae coordinates and angles. It is observed that one fingerprint can only contain limited number of minutia points,



**Figure 3.** Insert chaff points evenly.

usually 20–40. If attackers acquire minutiae information successfully, they can narrow down the search range of  $K_{\text{bio}}$ . In other words, the cryptographic key space created by 20–40 minutiae points is small.

There exist two solutions to solve this problem. One is increasing the total number of minutiae. The other is increasing the difficulty for attackers to capture the minutiae information. In our protocol, the second solution is adopted. We insert a large amount of chaff points (fake minutiae) into a template  $T$  so that genuine minutiae and fake minutiae are mixed, yielding the lookup table. In our protocol, the number of chaff points is greater than 200. A mobile device only stores  $T$ , without any indication of which nodes are genuine.

The security strength of a lookup table is significantly determined by how we mix the inserted chaff points and genuine minutiae. When the overall point distribution trend to be uniform, the maximum information entropy is achieved, i.e. the template is with strongest security. To achieve this goal, we partition a fingerprint area into several small cells, as illustrated in Figure 3. We insert a certain number of chaff points into each cell ensuring that all cells contain same number of points (chaff and genuine). Therefore, the probability of each point being guessed out is the same, yielding the maximum randomness (security strength) of a template.

Suppose a fingerprint of a mobile user  $U'$  has 30 minutiae. The number of chaff minutiae is 300. Authentication threshold  $n_{\text{threshold}}$  is 8 minutiae. The list  $L'$  contains 14 minutiae index numbers. The list can provide

$$C_{14}^{330} = 8 \times 10^{21} \text{ combinations} \quad (9)$$

To extract genuine minutiae from  $T$ , it is necessary to try all combinations. Suppose verifying one combination costs 5 s.  $8 \times 10^{21}$  combinations will take  $4 \times 10^{22}$  s, around  $1.2 \times 10^{15}$  years. As the authentication is conducted in the server side, the number of allowable malicious attempts on such combinations will be further limited as a server can easily detect the anomaly of authentication attempts from the same ID.

If an attacker randomly selects 14 points among all 330 points, the probability he passes the authentication test can

be calculated as

$$P = \frac{\sum_{i=8}^{14} C_i^{330} \cdot C_{(14-i)}^{330}}{C_{14}^{330}} = 5.86 \times 10^{-5} \quad (10)$$

The Equation (9) and (10) indicate inserting chaff points can effectively prevent attackers obtaining the original biometric data and reduce the probability of illegally gaining access to the system. Thus, it is hard to break the system from biometric template.

### 3.4. Transmission channel attack

The authentication messages exchange between  $U'$  and  $S_{\text{cen}}$  should be protected against channel attack since the broadcasting nature of mobile network. PKI can provide sufficient protections in terms of information confidentiality and integrity. The private keys ensure only the intended receivers are able to retrieve the original messages from ciphertext. The ECC algorithm employed in our protocol offers high security level without slowing down the system speed and consuming much resource. The 256-bit ECC have the same security level as 3072-bit RSA algorithm which is commonly used in current Internet.

### 3.5. Replay attack and man-in-the-middle attack

Replay attack is prevented by the adoption of timestamps in our protocol. The system is breakable only when an attacker has obtained the private key as well as previous transmitted messages. The  $L'$  will be broken. Even when this happens, the damage to future communication is limited. This is because there are many different  $L'$  lists. Repetition of the same  $L'$  or a couple of  $L'$  s will be easily detected by the central server, which can trigger an alarm of security compromise. We can also produce a fuzzy template  $T$  based on cancelable template technology. This will make it infeasible to derive genuine minutiae from  $L'$  list, which also renders  $L'$  list cancelable.

MITM (man-in-the-middle) attack refers to active eavesdropping where attackers make independent connections and relays messages between legal users. The communication is controlled by attackers while legal users believe that they are talking directly to each other over a private connection. Traditionally, MITM attack can be prevented by using PKI, because in PKI the public keys with respective user identities are bound and stored in a certificate authority (CA). Before conducting encryption, each user is able to verify the public key with CA to ensure it is not from a malicious server.

In addition, in the proposed protocol minutia index numbers are transmitted without any information leakage. Even if the PKI (public key infrastructure) has been compromised, the worst case is that malicious server could only obtain the transmitted index number. The final bio-key, used for client-server communication, relies on both index number and fuzzy template  $T$ . It means index numbers will be useless if without the possession of the registered fuzzy fingerprint template stored locally inside the client's mobile device and the server. Our protocol does not require to transmit template  $T$  hence it is impossible to obtain  $T$  using MITM attack. The protocol provides strong security strength.

## 4. EXPERIMENT

The experiment of our proposed protocol consists of two stages. At first, the fingerprint verification algorithm used in our protocol was implemented in Matlab and tested on a PC with public domain database. The main purpose is to estimate the performance of authentication accuracy. In stage two, we evaluate the ECC scheme on Java ME emulator environment in order to investigate its feasibility, resource demand, and computational speed.

### 4.1. Verification accuracy

The fingerprint sample sets we tested are from the NIST Special Database 24 [23] and FVC 2002 DB2 Database [24].

#### 4.1.1. Experiment on NIST 24.

In this experiment, a subset of five finger subjects was chosen from the distortion set each having 150 different impressions. This results in 750 individual fingerprints in our experiments. In NIST Special Database 24, the live-scanned fingerprints are obtained from an optical scanner of resolution 500 dpi with original dimension of  $720 \times 480$  pixels. For convenience, the selected fingerprints were first properly cropped, downsized *via* averaging and zero padding to a new size of  $256 \times 256$ .

We use four training impressions to generate one template. Firstly, we aligned training impressions according to their singular points. Then we take the union of minu-

tiae in four training impressions as the template  $T$ . In  $T$ , if one minutia  $m_i$  is too close to another  $m_j$ , i.e. the distance  $d(m_i, m_j) \leq d_{\text{threshold}}$  and  $\Delta\theta(m_i, m_j) \leq \theta_{\text{threshold}}$ ,  $m_i$  and  $m_j$  will be merged to a new minutia point  $m_k$ , where  $m_k$  is defined as  $\left(\frac{(x_i+x_j)}{2}, \frac{(y_i+y_j)}{2}, \theta_i\right)$ . On the other hand, we require that a user imprint the finger three times for authentication. Three test impressions will be combined to generate one test minutiae set  $Q$ . The procedure of minutiae combination is different from the procedure of generating template. Here we cross match the three query impressions to find all matched minutiae.  $Q$  only includes the matched minutiae so unmatched minutiae are discarded. This aim to ensure only the reliable minutiae can participate in the following matching process.

For each subject, 100 templates were generated *via* selecting different combinations of training images. For each template, we randomly selected 40 impressions from the same subject as genuine test and 160 impressions from the rest four subjects (40 impressions from each) as imposter test. Therefore, a total of  $100 \times 40 \times 5 = 20\,000$  genuine tests and  $100 \times 160 \times 5 = 80\,000$  imposter tests were performed.

The authentication accuracy of a biometric system can be assessed by several metrics. Two commonly used ones are false acceptance rate (FAR) and false rejection rate (FRR). Traditionally, FAR is defined as the ratio of the number of incorrectly accepted impostor tests to the total number of impostor tests. FRR is the ration of the number of incorrectly rejected genuine tests to the total number of genuine tests. Mobile device needs to make a selection from all matched minutiae. As  $L_0$  is bigger than  $L'$ , there exist many possible  $L'$  lists derived from  $L_0$ . From these combinations the numbers of false acceptance authentication test results and false rejection authentication test results can be calculated. For each matching result, we calculate the probability of a genuine user/imposter passes the authentication test. We redefine the FAR and FRR as:

$$\text{FAR} = \frac{\sum \text{Probability of a impostor passes a test}}{\text{Total number of impostor tests}} \quad (11)$$

$$\text{FRR} = \frac{\sum \text{Probability of a genuine test be rejected}}{\text{Total number of genuine tests}} \quad (12)$$

Figure 4 plots a set of FAR–FRR pairs produced by the authentication scheme of the proposed protocol with different matching threshold. As the threshold increases, the FAR decreases while FRR increases. The three curves describe the situation that 10, 12, 14, 16 minutiae are selected and sent to the central server. As can be seen, the more minutiae selected, the better authentication performance achieved.

Since the total number of minutiae varies from one finger to another, different fingerprint may show different performance. Thus, in addition to overall system performance curve, we also conduct tests for each subject, respectively. The ROC curves are as shown in figures.

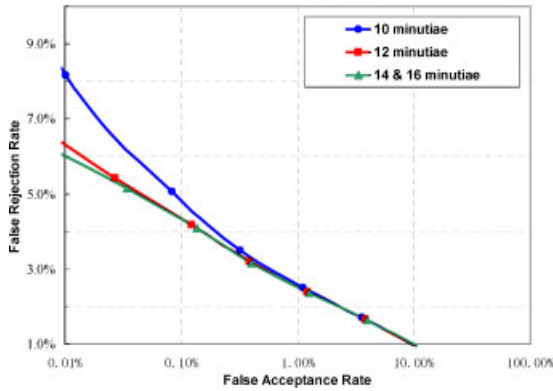


Figure 4. ROC Curves of different number of selected minutiae.

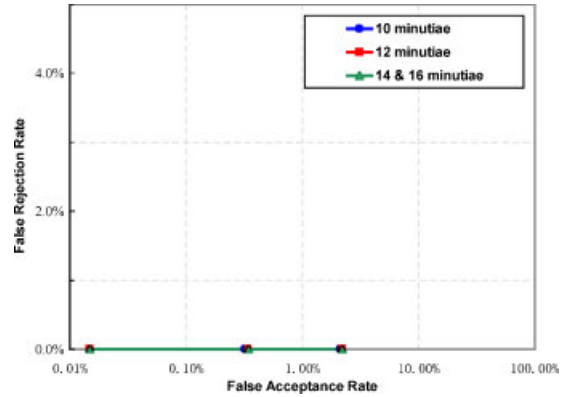


Figure 7. ROC curves of the subject 'fm\_pm3\_5'.

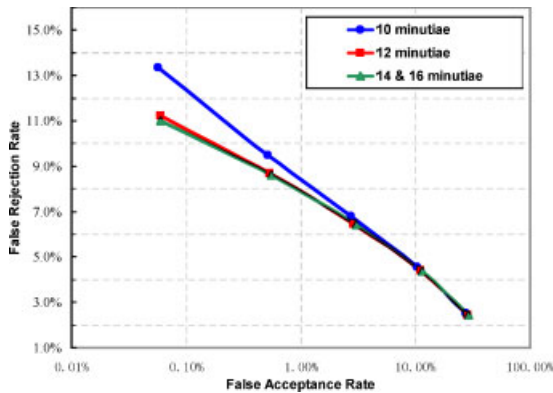


Figure 5. ROC curves of the subject 'fm\_pm0\_5'.

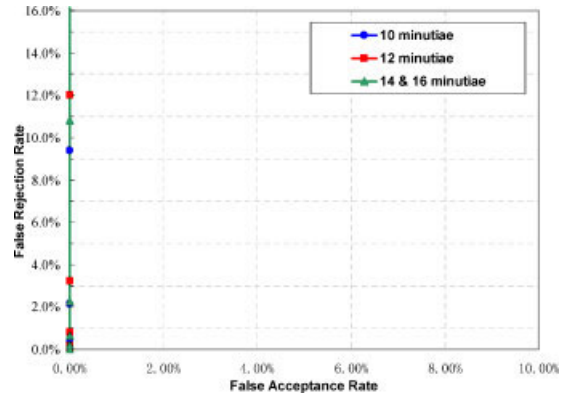


Figure 8. ROC curves of the subject 'fm\_pm4\_5'.

Figures 5–9 indicate that the performance improves when the number of selected minutiae increases. We consider two situations. One is when FAR and FRR tend to be equal and one is when FAR = 0.01%. The results are summarized in Table I.

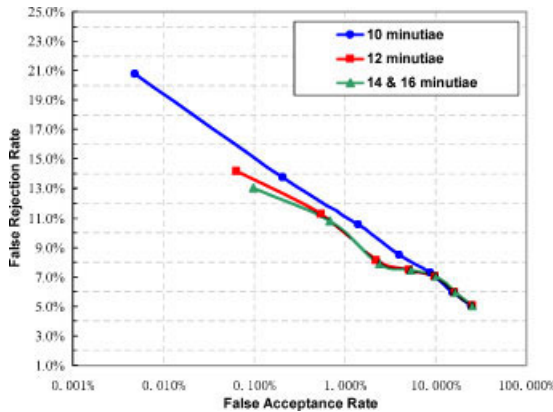


Figure 6. ROC curves of the subject 'fm\_pm2\_5'.

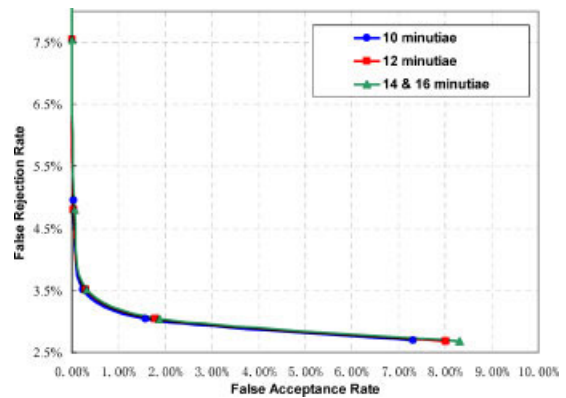


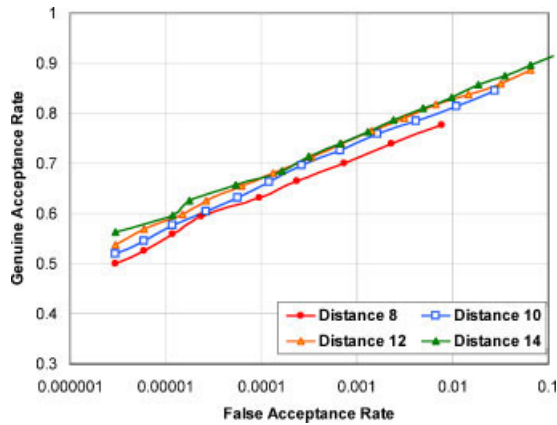
Figure 9. ROC curves of the subject 'fm\_pm5\_5'.

#### 4.1.2. Experiment on FVC 2002.

FVC2002-DB2 contains 800 live-scanned fingerprints (100 fingers each give 8 different impressions) in total. All images are captured by an optical sensor with a resolution of 500 dpi. In the experiment we chose impression NO.1, 2, 7, 8 for our experiments. Impression 3, 4, 5, and 6 in this database were obtained by requesting users to provide fingerprints with exaggerated displacement and rotation [25]. It makes sense that the users are willing to provide good

**Table I.** Matching performance for different subjects

| Subject  | Number of select minutiae | EER (FAR = FRR) % | FRR when FAR = 0.01 % % |
|----------|---------------------------|-------------------|-------------------------|
| fm_pm0.5 | 14                        | 5.4               | 10.9                    |
| fm_pm2.5 | 14                        | 7.2               | 13                      |
| fm_pm3.5 | 14                        | 0                 | 0                       |
| fm_pm4.5 | 14                        | 0                 | 0                       |
| fm_pm5.5 | 14                        | 2.7               | 4.7                     |

**Figure 10.** Performance result on FVC 2002 database.

quality biometric information to the server. Hence, it is reasonable that impression 3, 4, 5, and 6 were not considered for our experiment.

The experimental process is slightly different from the one we conducted on NIST 24. This time we tested the GAR against FAR under different value of  $d_{\text{threshold}}$ . For one subject, two impressions were combined to generate a template and the other two impressions are for a test. There were 3454 genuine tests and 337 602 imposter tests in total. The minutiae combination process was conducted with the help of commercial fingerprint software Verifinger 5.0, in order to achieve the best registration effect. When generating either template or test sample, verifinger can help align two training impressions. In real application, this alignment operation can be done by hardware, for instance UPEK TCS5B [17].

In Figure 10, the genuine acceptance rate is plotted against false acceptance rate. The four curves are corresponding to the situation that  $d_{\text{threshold}} = 8$ ,  $d_{\text{threshold}} = 10$ ,  $d_{\text{threshold}} = 12$ , and  $d_{\text{threshold}} = 14$ , respectively. In theory, increasing the threshold  $d_{\text{threshold}}$  would result in that more genuine users and imposters are accepted by the system, yielding high GAR and FAR. However, in our experiment higher FAR were not observed when bigger  $d_{\text{threshold}}$  values were used. We believe that it is due to the usage of multiple impressions. The process of combining (find intersection) two impressions, in fact, eliminates the specious minutiae and significantly reduces the probability of the specious minutiae randomly match the template.

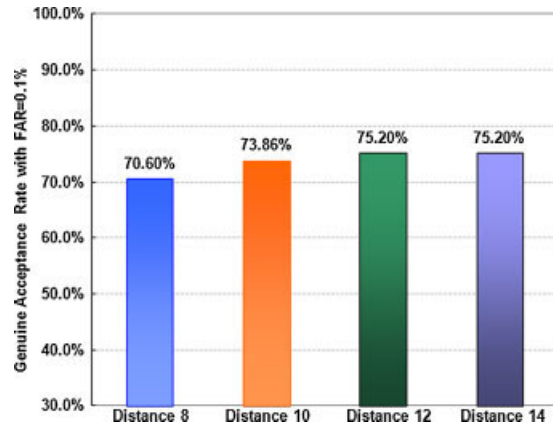
**Figure 11.** GAR with FAR = 0.1%.

Figure 11 demonstrates the GAR at FAR = 0.1%. GAR = 78.69% is achieved when  $d_{\text{threshold}} = 14$ . We believe the performance is at an acceptable level. Better authentication accuracy can be achieved when more sophisticated image registration and matching algorithms are employed.

#### 4.2. Resource demand and speed in Java ME

Java Platform, Micro Edition (Java ME) is one of most popular mobile application development technology designed for mobile devices such as mobile phones. A major advantage of Java ME is its cross-platform nature which means the same source code can be executed on all platforms without modification.

The Java ME platform has two versions [26], one for general mobile devices, named the Connected Limited Device Configuration (CLDC), and the other one for more capable mobile devices like smart-phones and set top boxes, named the Connected Device Profile (CDC). Our application is deployed based on CLDC because CLDC is more popular and is widely supported by nearly all Java-enabled mobile phones.

We were using Sun Java Wireless Toolkit (WTK) 2.5 emulator as the development and test platform. WTK provides a perfect support to the latest Java ME CLDC technology and Mobile Information Device Profile (MIDP 2.0).

**Table II.** Computational time of messages with different lengths

|                                    | Java ME<br>(mobile-end<br>application) | Java SE<br>(server-end<br>application) |
|------------------------------------|--|--|
| ECC Encryption 256 bit<br>message  | 1.36 s                                 | 0.115 s                                |
| ECC Encryption 2700<br>bit message | 12.45 s                                | 1.33 s                                 |
| ECC Decryption 256 bit<br>message  | 0.46 s                                 | 0.042 s                                |
| ECC Decryption 2700<br>bit message | 8.21 s                                 | 0.743 s                                |

We implemented the ECC from the scratch. A few basic functions in the open-source cryptographic libraries [27] have been utilized, to handle cryptography-related operations such as mathematical operations over the Galois Field. The domain parameter of ECC we used is *secp256v1* which is specified in Reference [28]. One template is allowed to have up to 1024 minutiae therefore the minutiae index number can be represented by a 10-bit stream. If 16 minutiae are selected and transmitted to central servers, the length of a message is 256 bits (160 bit minutiae information, 64 bit user ID, and 32 bit time stamp). Note that public-key cryptography is sensitive to the message length. In real application, public-key algorithm is only applied in encrypting short-length messages. Transmitting the proposed bio-information (minutiae index number, user ID, and time stamp) is more computation efficient than raw minutiae information. In our experiment we compared the computational speed of encrypting/decrypting 256 bit message with the speed of operating on 2700 bits (raw minutiae information), as shown below. Besides Java ME, Java SE implementation has also been tested, as the simulation of encryption/decryption process on the server-side.

In the Table II, it is apparent that encrypting/decrypting 2700 bit message consumes over 10 times longer time than 256 bit message. Assume in a real scenario that a server with 1000 simultaneous incoming requests, the processing time of handle 2700 bit message will be  $0.743 \text{ s} \times 1000 = 743 \text{ s} \approx 13 \text{ min}$ , which would likely incur a system timeout and show no-response to mobile client. If using the proposed 256 bit message, the processing time will be  $0.042 \text{ s} \times 1000 = 42 \text{ s}$ , which become quite acceptable.

In addition, average memory usage is 1.14 MB where most recent mobile phones support more than 10 MB memory. It can be seen that the computational speed and resource demand are within acceptable range for both mobile client and server.

## 5. CONCLUSIONS

We have presented a bio-cryptographic security protocol designed for client-server authentication in mobile computing environment. Different from existing key binding

and key release schemes, user authentication of our protocol is conducted remotely at the server side. The proposed scheme assumes that the server is secure which is a common practice in most client-server applications. In order to handle different types of attacks, we designed or adopted different security mechanisms: ECC PKI is used to protect authentication process. Fuzzy template is proposed to secure the genuine biometric feature against attacks. The bio-key, dynamically generated from fingerprint, is designed to protect communication between client and server after successful authentication. Performance evaluation on NIST 24 database shows a reasonable matching accuracy has been achieved. The implementation on Java ME emulator proves the proposed protocol satisfies the resource-constrained mobile devices. Future work includes adopting more sophisticated fingerprint matching algorithm and cancelable biometric template protection schemes.

## ACKNOWLEDGEMENTS

This paper is support by ARC (Australia Research Council) Discovery Grant DP0985838.

## REFERENCES

1. Shaw K. Data on PDAs mostly unprotected. Network World Fusion. Available from [www.nwfusion.com](http://www.nwfusion.com) 2004.
2. Rivest RL, Shamir A, Adleman L. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM* 1978; **21**(2): 120–126.
3. Koblitz N. Elliptic curve cryptosystems, *Mathematics of Computation* 48, 1987; 203–209.
4. Miller V. Use of elliptic curves in cryptography, *CRYPTO* 85, 1985.
5. Maltoni D, Maio D, Jain AK, Prabhakar S. Handbook of Fingerprint Recognition. Springer-Verlag: New York, 2003.
6. Soutar C, Roberge D, Stojanov SA, Gilroy R, Vijaya Kumar BVK. Biometric encryption - enrollment and verification procedures. *Proceedings of SPIE, Optical Pattern Recognition IX* 1998; **3386**: 24–35.
7. Soutar C, Roberge D, Stojanov SA, Gilroy R, Vijaya Kumar BVK. Biometric encryption using image processing. *Proceedings of SPIE, Optical Security and Counterfeit Deterrence Techniques II*, 1998; **3314**: 178–188.
8. Soutar C, Roberge D, Stojanov SA, Gilroy R, Vijaya Kumar BVK. Biometric encryption. In *ICSA Guide to Cryptography* Nichols RK (ed.). McGraw Hill, New York, 1999.
9. Teoh A, Goh A, Ngo D. Random multispace quantization as an analytic mechanism for bihashing of biometric and random identity inputs. *IEEE Transactions on Pat-*

- tern Analysis and Machine Intelligence* 2006; **28**(12): 1892–1901.
10. Savvides M, Vijayakumar B. Cancellable Biometric Filters for Face Recognition. *Proceedings of IEEE International Conference Pattern Recognition*, volume 3, pages 922–925, Cambridge, UK, August 2004.
  11. Ratha N, Chikkerur S, Connell J, Bolle R. Generating Cancelable Fingerprint Templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 2007; **29**(4): 561–572.
  12. Teoh A, Toh K, Yip W. 2N Discretisation of BioPhasor in Cancellable Biometrics, *Proceedings of Second International Conference on Biometrics*, Seoul, South Korea, 2007; 435–444.
  13. Dodis Y, Ostrovsky R, Reyzin L, Smith A. Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. *SIAM Journal of Computing* 2008; **38**(1): 97–139.
  14. Juels A, Sudan M. A fuzzy vault scheme. In Lapidath A, Teletar E (eds). *Proceedings of IEEE International Symposium on Information Theory*, 2002; 408.
  15. Uludag U, Pankanti S, Jain AK. Fuzzy vault for fingerprints, *Proceedings of Audio- and Video-based Biometric Person Authentication*. Rye Town: USA, 2005; 310–319.
  16. Xi K, Hu J. Biometric mobile template protection: a composite feature based fingerprint fuzzy vault, *IEEE International Conference on Communications*, Dresden, Germany, 2009.
  17. [www.upek.com/solutions/physical/chipsets\\_sensors.asp](http://www.upek.com/solutions/physical/chipsets_sensors.asp)
  18. [www.authentec.com/products-accesscontrol-aes2510.cfm](http://www.authentec.com/products-accesscontrol-aes2510.cfm)
  19. [avnetexpress.avnet.com/store/em/EMController/Sensors-and-Transducers-Misc/UPEK/TCS5BB6A0/\\_JR-9066904/A-9066904/An-0?action=part&catalogId=500201&langId=-1&storeId=500201](http://avnetexpress.avnet.com/store/em/EMController/Sensors-and-Transducers-Misc/UPEK/TCS5BB6A0/_JR-9066904/A-9066904/An-0?action=part&catalogId=500201&langId=-1&storeId=500201)
  20. [www.neurotechnology.com/verifinger.html](http://www.neurotechnology.com/verifinger.html)
  21. Wang Y, Hu J, Phillips D. A fingerprint orientation model based on 2D Fourier expansion (FOMFE) and its application to singular-point detection and fingerprint indexing. *IEEE Transactions on PAMI* 2007; **29**(4): 573–585.
  22. FIPS. 180-2: Secure Hash Standard (SHS) 25 February 2004.
  23. Watson. NIST special database 24, live-scan digital video fingerprint database. *Technical report, U.S. National Institute of Standards and Technology*, 1998.
  24. Maio D, Maltoni D, Cappelli R, Wayman JL, Jain AK. 'FV C2002: Second Fingerprint Verification Competition,' *ICPR*, vol. 3, pp. 30811, *16th International Conference on Pattern Recognition (ICPR'02)* Volume 3, 2002.
  25. Nandakumar K, Jain AK, Pankanti S. Fingerprint-based Fuzzy Vault: Implementation and Performance, *IEEE Transactions on Informatics Forensics and Security*, vol. 2, no. 4, pp. 744–757, December 2007.
  26. Sun Website, URL: [java.sun.com](http://java.sun.com) 2009.
  27. Bouncy Castle. *Lightweight API*, The Legion of the Bouncy Castle, 2008.
  28. Certicom. *SEC 2: Recommended Elliptic Curve Domain Parameters*, Certicom Corp., 2000.
  29. Ahmad T, Hu J, Han S. Efficient Mobile Voting System Security Scheme based on Elliptic Curve Cryptography, *International Workshop on Intelligent Decision Support Systems and Applications in Networked and Distributed Systems, IEEE 3rd International Conference on Network & System Security (NSS09)*, Gold Coast, Australia, October 2009; 19–21.
  30. Han F, Hu J, Yu X, Feng Y, Zhou J. A novel hybrid crypto-biometric authentication scheme for ATM based banking applications, *IAPR International Conference on Biometrics (ICB2006)*, Hong Kong China, 5–7 January, 2006. Published at *Lecture Notes in Computer Science*, Springer, vol. 3832/2005, 2005; 675–681.
  31. Mahmood A, Hu J, Tari Z, Leckie C. Critical infrastructure protection: Resource efficient sampling to improve detection of less frequent patterns in network traffic. *Journal of Network and Computer Applications*, Elsevier, 2010.
  32. Wang Y and Hu J. Global Ridge Orientation Modelling for Partial Fingerprint Identification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.
  33. Hu J, Chen H.H., Hou T.W. A hybrid public key infrastructure solution (HPKI) for HIPAA privacy/security regulations. *Special Issue on Information and Communications Security, Privacy and Trust: Standards and Regulations*. *Computer Standards & Interfaces*, Elsevier, 274–280.
  34. Hoang X.D, Hu J, Bertok P. A Program based anomaly intrusion detection scheme using multiple detection engines and fuzzy inference. *Journal of Network and Computer Applications*, Elsevier, 32 (2009) 1219–1228.
  35. Hu J, and Han F. A pixel-based scrambling scheme for digital medical images protection. *Journal of Network and Computer Applications*, Elsevier, 32(2009) 788–794.
  36. Hu J, Qiu D, Chen H.H., Yu X. A simple and efficient data processing scheme for HMM based anomaly intrusion detection. *Special Issue of Advances on Network Intrusion Detection*. *IEEE Network*, vol.23, no.1, January, 2009, pp.42–47.
  37. Hu J and Zambetta F. Security issues in massive online games, *Journal of Security and Communication Networks*, John Wiley, Issue 1, 2008, pp.83–92.

38. Wang Y, Hu J, Han F. Enhanced gradient-based algorithm for the estimation of fingerprint orientation field. *Applied Mathematics and Computation*, Elsevier, Vol. 185, No. 2, pp. 823-833, Feb 2007.
39. Han F, Hu J, Yu X, Wang Y. Fingerprint images encryption via multi-scroll chaotic attractors. *Applied Mathematics and Computation*, Elsevier, Vol. 185, pp.931-939, 2007.
40. Han F, Yu X, Feng Y, Hu J. On multi-scroll chaotic attractors in hysteresis-based piecewise linear systems. *IEEE Transactions on Circuits and Systems-II*, Vol. 54, Issue 11, Nov. 2007, pp. 1004-1008.