A NEW SECURE DESIGN FOR MOBILE COMMUNICATION

SEIFEDINE KADRY[1,2], HUSSAM KASSEM[2]

[1]Lebanese University, Faculty of Science - Lebanon

[2]Lebanese International University, Faculty of Engineering - Lebanon

Email: skadry@gmail.com

ABSTRACT

Mobile handheld device is a popular device that provides secure, private, authentic, and accurate communication and exchange of confidential information. In this paper we propose a technique to solve the authenticity problem in mobile communication. This technique is mainly based on the usage of the Fingerprint to identify both the speaker and the sender. This technique is simple, requires less calculation than other public/private key techniques, assures more authenticity than digital signature, and eliminates the need for a third party. Moreover, when applied to mobile phones, this technique resists any forge imposed by another party.

Key words: *Cryptography, Secure Telecommunication, Fingerprint Recognition.*

1. INTRODUCTION

A recent survey carried by IDC shows that around 90% of mobile users use messaging as their main communication tool disregarding the safety level of such a communication system; if phones are lost or shared, anyone can access the data on the phone. This is known as the AUTHENTICITY in cryptography science. That is why, scientists should come up with a concept that minimizes the risk associated with losing or sharing a phone, thus offering a safe environment for communication.
This paper presents a solution for the above mentioned problem. "Fingerprint Identification Technique" is the most effective technique for solving such a problem. This technique works on the Fingerprint basis whereby the phone can be accessed when it identifies the Fingerprint of the user(s).

This paper is organized as follows: In section 2, we provide an overview of the secure communication in mobile handheld device. Then, section 3 describes the digital signature scheme and the related algorithm RSA. In section 4, we write the code of the RSA algorithm in JAVA for the performance purpose. Section 5 gives a literature review of fingerprint matching technique. Next in section 6, we describe the proposed design which is based on the fingerprint to authenticate the caller, the performance of our design is given in section 7. Finally, in section 8 we conclude the paper with future work.

2. RELATED WORK

Digital signature can be represented as a secure base in such applications, because it provides authentication services. Traditional digital signature schemes are based on asymmetric cryptographic techniques, which make the signature computation very expensive. Although handheld devices are of many shapes and can be used for different purposes, they have some limitations; they have a limited computational capability and a short battery life.
There are many proposed digital signature schemes in literature. According to their bases, these schemes can be classified into two general categories: message digest based schemes and message recovery based schemes. In message digest based digital signature scheme, the original message is first mapped to a checksum, which is used to provide data integrity, by a one-way function. Then this checksum (message digest) is used to generate a digital signature. On the other hand, in message recovery based scheme, the receiver can recover the original message from the received signature that is performed by the message redundancy scheme. There also exists some work/theory on digital signatures for mobile devices. For instance Asokan *et al.* [1] proposed Server-Supported Signature scheme for mobile

communication. Their work employed a one-way function and traditional digital signature scheme. Signature servers were responsible for generating signature tokens and certification authorities to verify these tokens. Therefore, the scheme's robustness depends on the reliability of those servers. Based on the work of Asokan *et al.* [1], Ding *et al.* [2] presented a modified digital signature scheme, called Server Aided Signature. In this scheme, users are involved in the generation of the signature token, given the on-line feature and defending the DOS attack. Unlike the traditional digital signatures which often use the pair public/secret keys to generate non-reputation signature token, [1] and [2] use one way hash function to generate sender's secret key and use this key to produce non reputation signature token. This is due to the fact that generation of robust pair public/secret keys is computation intensive for handheld devices. But in their schemes, asymmetric cryptographic computations are still expensive for clients because during the period of verification of digital signature, clients should decrypt the encrypted signature token to verify the associated content added by signature servers.

3. RSA AND DIGITAL SIGNATURE

Since the "digital signature schemes" is based on the RSA algorithm, it is important to introduce and simplify the principle of RSA, which is probably the most recognizable asymmetric algorithm. RSA was created by Ron Rivest, Adi Shamir, and Leonard Adleman in 1978 [3]. Till now, it is the only asymmetric (i.e. needs two different keys) algorithm used for private/public key generation and encryption. The operation of the RSA is as follows:

1. Choose two distinct large random prime numbers *p* and *q*
2. Compute $n = p\,q$, where *n* is used as the modulus for both the public and private keys
3. Compute the totient: $\varphi(n) = (p-1)(q-1)$
4. Choose an integer *e* such that $1 < e < \varphi(n)$, and *e* and $\varphi(n)$ share no factors other than 1 (i.e. *e* and $\varphi(n)$ are coprime), where *e* is released as the public key exponent
5. Compute *d* to satisfy the congruence relation $d \times e = 1$ modulus $\varphi(n)$; *d* is kept as the private key exponent

Two mathematical problems exist in the RSA cryptosystem [4]: the problem of factoring very large numbers, and the RSA problem. The integer factorization problem deals with finding a nontrivial factor of a composite almost prime number. When these numbers are very large, it becomes difficult to factorize. Whereas, the RSA problem is simply the task of taking *e*-th roots modulo a composite *n*. Trying to get the plaintext *m* such that $m\,e = c \mod n$, where the RSA public key is *e* and *n*. An attacker needs to factor *n* into *p* and *q*, and compute $(p-1)(q-1)$ that allows the determination of *d* from *e*.

Digital signature provides authenticity by using RSA algorithm, i.e. by using public and private keys. To sign a message, Alice encrypts the message using her private key. To verify the signature, Bob looks up the Alice's public key and uses it to decrypt the message. Since only Alice knows the private key, only Alice can decrypt a message that can be decoded with the public key. The following picture explains this process graphically (fig. 1).

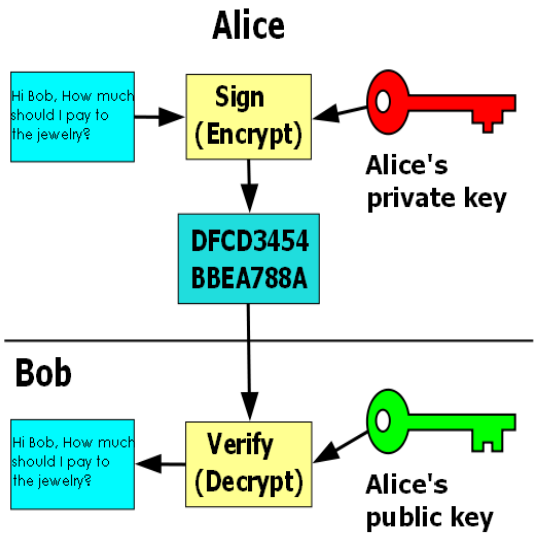


Figure 1: Digital Signature Process

4. RSA in JAVA

For the performance analysis, we have written the RSA algorithm in JAVA to be easily tested in a mobile with Symbian Operating System.

```
import java.security.KeyPair;
import java.security.KeyPairGenerator;
import java.security.PrivateKey;
import java.security.PublicKey;
import javax.crypto.Cipher;
class encryptor{
private PrivateKey privateKey;
private PublicKey publicKey;
private String encryptionType;
public encryptor(int encryptionStrength){
encryptionType = "RSA";
try {
KeyPairGenerator kpg;
kpg =
KeyPairGenerator.getInstance(encryptionType);
kpg.initialize(encryptionStrength);
KeyPair myPair = kpg.generateKeyPair();
publicKey = myPair.getPublic();
privateKey = myPair.getPrivate();
}catch (Exception e){
e.printStackTrace();
}
}
public PublicKey getPublicKey(){
return publicKey;
}
public byte[] encrypt(byte[]
unEncryptedByteArray, PublicKey key){
try {
byte[] cipherText = null;
Cipher cipher =
Cipher.getInstance(encryptionType);
cipher.init(Cipher.ENCRYPT_MODE, key);
cipherText =
cipher.doFinal(unEncryptedByteArray);
return cipherText;
} catch (Exception e) {
e.printStackTrace();
return null;
}
}
public byte[] decrypt(byte[] encryptedByteArray){
try {
byte[] dectyptedText = null;
Cipher cipher =
Cipher.getInstance(encryptionType);
cipher.init(Cipher.DECRYPT_MODE,
privateKey);
dectyptedText =
cipher.doFinal(encryptedByteArray);
return dectyptedText;
} catch (Exception e) {
e.printStackTrace();
return null;
}
}
}
public class testEncryptor {
public static void main(String[] args) {
String str = "Hello Alice, What are you doing
tonight?";
byte[] text = str.getBytes();
System.out.println( new String(text) + ",
length:"+text.length);
encryptor encryp = new encryptor(2048);
System.out.println( "key made:
"+encryp.getPublicKey().toString());
long start = System.currentTimeMillis();
byte[] encryptedByteArray = encryp.encrypt(text,
encryp.getPublicKey());
System.out.println(new
String(encryptedByteArray));
long finish = System.currentTimeMillis();
System.out.println("execution time to encrypt:" +
(finish - start) + " millis to exec");
long start1 = System.currentTimeMillis();
byte[] decryptedByteArray =
encryp.decrypt(encryptedByteArray);
System.out.println(new
String(decryptedByteArray));
long finish1 = System.currentTimeMillis();
System.out.println("execution time to decrypt:" +
(finish1 - start1) + " millis to exec");
}
}
```

5. OVERVIEW OF FINGER PRINT MATCHING ALGORITHM

Fingerprint verification is a quick, convenient, and effective method of establishing an individual's identity. Among all the biometric techniques, Fingerprint-based identification is the oldest. By definition, a Fingerprint is a series of three-dimensional lines, ridges, and the spaces between them, valleys. Features found in the unique pattern of a Fingerprint's ridges and valleys are involved in the verification of an identity. Anatomic characteristics called minutiae are the locations on a Fingerprint where the ridges begin, stop, fork, or intersect. Minutia extraction analyzes and identifies the key features of the Fingerprint such as the location and direction of the ridges. Some approaches use only minutiae for matching, while others include information such as the number of ridgelines between adjacent minutiae (fig. 2).

Figure 2: Minutiae

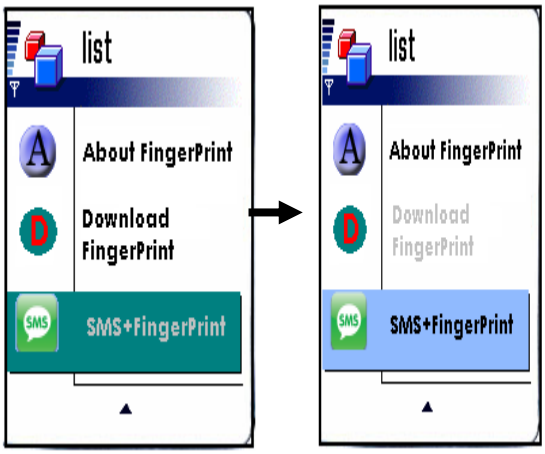
When the Fingerprint image is analyzed, the minutiae points are extracted and translated into a code that serves as a template. Templates usually have a size of between 40 and 1000 bytes, often around 256 bytes [5].

The algorithms used to resolve Fingerprint matching can be classified [6] into three main branches: *correlation-based* [7], where the match is performed by superimposing two Fingerprint images and computing the correlation between corresponding pixels (in this case the template is directly the finger image); *minutiae-based*, whose theory is fundamentally the same as for manual Fingerprint examiners, and *ridge feature-based* [8], where the Fingerprints are compared in terms of ridge pattern features other than the minutiae or pixels intensity, like texture information or sweat pores. Focusing on the minutiae based algorithms, the match procedure essentially consists in finding the maximum number of corresponding minutiae between two templates; this problem can be addressed also as a more general *point pattern matching* problem [9].

6. THE PROPOSED DESIGN

As mentioned in the previous sections, the existing solutions to solve the authenticity problem are based on the calculation of the public and private keys. The main drawback of this solution is mainly related to its performance in terms of the execution time in the encryption and decryption and the limitation of the mobile device. To solve this drawback, our design is based on the Fingerprint process. The scenario is as follows:

1- When a person buys a SIM Card from a seller, his or her Fingerprint should be taken and send to the database of the service provider and saved beside its number.
2- New options should be added to the mobile system; for instance talking with Fingerprint (fig. 3b) or sending messages with Fingerprint (fig. 3a) is a way to assure the authenticity.
3- The first step in using this system is downloading the user's Fingerprint form the service provider database and save it to the SIM Card. It is important to note here that in case of reselling the SIM Card, the Fingerprint can be removed.
4- When the caller/sender uses these options to authenticate him/herself for the receiver, the mobile system asks for the Fingerprint that matches with the one on the SIM Card. If the matching succeeds, a true symbol is added to the number and appears on the receiver mobile device to prove the authenticity of the caller or the sender (fig .4).



First time use after downloading

Figure 3a: SMS with Fingerprint Process

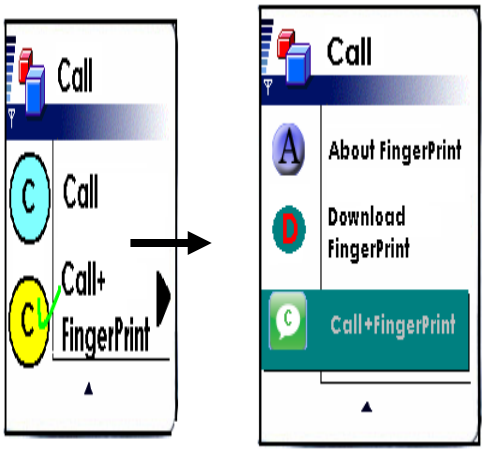


Figure 3b: Call with Fingerprint Process

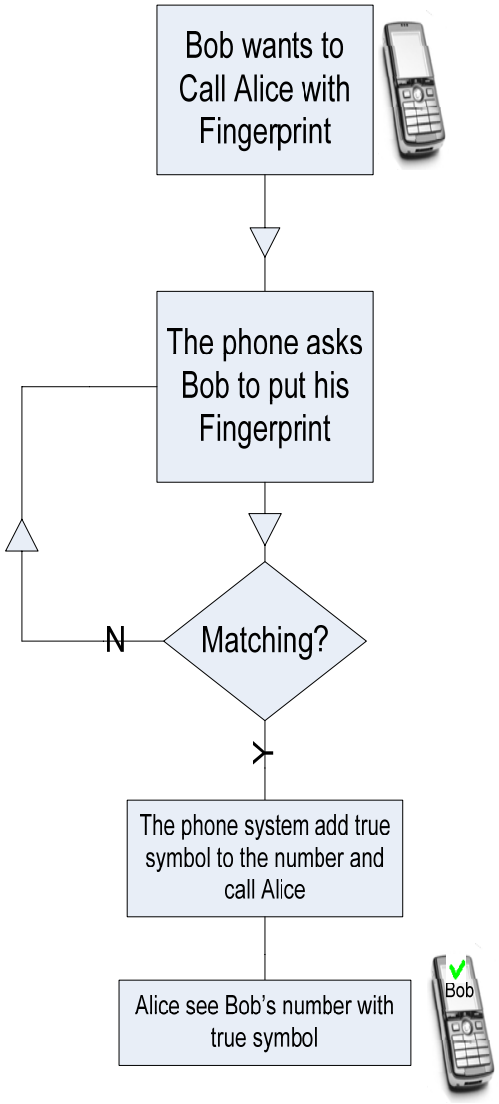


Figure 4: Flowchart of the Proposed Design

7. PERFORMANCE ANALYSIS OF THE PROPOSED DESIGN

In this section, the performance of the authenticity identification using Fingerprint with the existing solution, i.e. using private and public keys which is based on RSA algorithm, are compared. The comparison between these two techniques is made on the following CPU's clock: 60, 233 and 800MHz with RAM and ROM are equal to 128 MB. The results are shown in the following table:

Table 1: Execution Time in Seconds

	60MHz	233MHz	800MHz
RSA (encryption + decryption)	3.38 s	0.9	0.3
Fingerprint	1 s	0.26	0.075

Our experiment used the Asymmetric Fingerprint algorithm [10] for the Fingerprint matching. For RSA, we disregarded the time needed to generate the two prime numbers since they are generated only once.

It is important to note here that our system has several advantages:

1- Our design needs less time than the existing solution that is based on the RSA algorithm
2- Our system does not need a third party to verify the authenticity
3- Our system is more secure than RSA especially in the case of phone sharing or loss.

8. CONCLUSION AND FUTURE WORK

This paper introduces a new design to solve the authenticity problem in mobile communication. The above analysis, with experimental result support, shows that the proposed design is of a better performance and security than public/private keys technique which used RSA algorithm. We are working on implementing our design studying other security problems like privacy and confidentiality. Moreover, the extraction or composure of a key from the minutiae of Fingerprint is going to be studied as well.

REFERENCES:

[1] N. Asokan, G. Tsudik, M. Waidner. Server-supported signatures. Journal of Computer Security, Volume 5, Issue 1, pages 91–108, January 1997.

[2] Xuhua Ding, Daniele Mazzocchi, Gene Tsudik. Experimenting with Server-Aided Signatures. In Proceedings of Network and Distributed System Security Symposium (NDSS'2002), San Diego, 2002.

[3] Ronald L. Rivest, Adi Shamir, Leonard M. Adleman: A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Commun. ACM 21(2): 120-126 (1978).

[4] William Stallings. 2003. Cryptography and Network Security: Principles and Practice, Third edition. U.S. Prentice Hall.

[5] Nicky Boertien, Eric Middelkoop, Authentication in Mobile Applications, CMG, Telematica Instituut, The Netherlands, January 2002, <URL: https://doc.telin.nl/dscgi/ds.py/Get/File-23314/VH_authenticatie.pdf>.

[6] D. Maltoni, D. Maio, A.K. Jain, S. Prabhakar, Handbook of Fingerprint Recognition, Springer, 2003, ISBN 0-387-95431-7.

[7] T. Hatano , T. Adachi, S. Shigematsu, H. Morimura, S. Onishi, Y. Okazaki, H. Kyuragi, A Fingerprint Verification Algorithm Using the Differential Matching Rate, ICPR02, III volume: pp. 799-802, 2002.

[8] A. K. Jain, S. Prabhakar, L. Hong and S. Pankanti, Filterbank-based Fingerprint Matching, IEEE Transactions on Image Processing, Vol. 9, No.5, pp. 846-859, 2000.

[9] S. Bistarelli, G. Boffi, F. Rossi, Computer Algebra for Fingerprint Matching, Proc. International Workshop CASA'2003, Springer LNCS vol. 2657 2003.

[10] Stefano Bistarelli1,2, Francesco Santini2, and Anna Vaccarelli2, An asymmetric fingerprint matching algorithm for Java Card Pattern Analysis & Applications archive Volume 9 , Issue 4 (October 2006) table of contents Pages: 359 - 376, Year of Publication: 2006. ISSN:1433-7541.