Propose a Model for Securing SMS

Dr.Shaimaa H. Shaker, Dr.Hassan A. Jeiad, Fatimah A. Hassan

Abstract— The short message service (SMS) is one of the highly used mobile services for communication between parties. Each message can contain at most 140 bytes (1120 bits) of data, the equivalent of up to 160 English characters, or 70 Arabic or Chinese characters. This paper reviews the SMS architecture, security threats during its transmission and a proposed model of SMS encryption and authentication. It involves suitable encryption and decryption Algorithms using asymmetric cryptography and suitable HMAC algorithm as a message digest .The proposed model provides the confidentiality, integrity and authentication using modified RSA-2048 and SHA-256.

Index Terms— SMS, SMS Architecture, SMS Operation, SMS Security threats, SMS Proposed model, HMAC, RSA-2048.

1 INTRODUCTION

THE Short Message Service Center (SMSC) is the telecommunication operator that is responsible to rout and deliver the SMS to its correct destination, it is based on store-and-forward mechanism, SMS is stored as plain text in the SMSC, SMS is usually used to transmit unclassified information but nowadays with the rise of mobile commerce it became a powerful tool to transmit sensitive information. SMSC employees who have access to the network can read and modify the content of the messages, if an attacker compromises the SMSC he/she will also be able to see the traffic of the messages going through the SMSC. SMS doesn't provide security services by default; it is the job of the mobile communication systems to provide the protection needed [1]-[2].

There are four basic security requirements for the mobile networks: Authentication prevents unauthorized user from accessing services. Confidentiality protects sensitive data against eavesdropping. Integrity protects sensitive data from tampering and modification. Non-repudiation prevents forgery in network services. Encrypting the message will ensure its confidentiality and integrity. Hashing the message provides message authenticity, hashing is the transformation of a string of characters into a usually shorter fixed-length value or key that represents the original string. Asymmetric cryptography's main idea is that the public key is used to provide protection to the distribution of the private key that is used to secure SMS during its transmission [3]. This paper Introduce a model that used several cryptography techniques to overcome the SMS security problems and fulfill the security requirements over public network.

Some challenges are listed in GS1_mobile_com [4]. A hybrid compression encryption technique to secure the SMS data is proposed by Tarek M Mahmoud [5]. The computational cost overhead that the security protocols and algorithms impose on mobile phones is analyzed by Anita & Nupur Prakash [6]. Johnny Li-Chang Lo etal, proposed a two-phase protocol with the first handshake which occurs only once and uses asymmetric cryptography, and an n^{th} handshake which is more efficient symmetric handshake that is used frequently [7]. The main objective of this paper is to present authenticate encrypted short message of mobile using model consists of some cryptographic algorithms with key generator algorithm to ensure the security of SMS. The rest of this paper is organized as follows: In section 2 Short Message Service Architecture and Operation is explained. Section 3 discusses the proposed model for SMS security. The implementation of the proposed model is explained in section 4. In section 5 results are discussed. Finally, Section 6 concludes the paper.

2 SHORT MESSAGE SERVICE

This section presents an overview of the SMS operation, architecture and security threats.

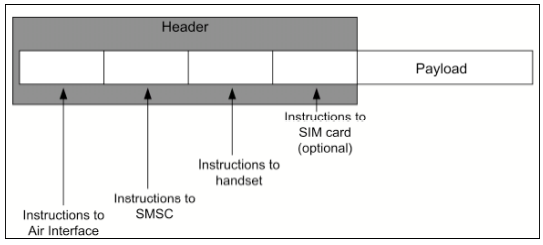

Fig. 1. SMS Packet Structure

2.1 SMS Architecture and Operation

The SMS packet Fig. 1 can go through in both directions, thus, when a message is sent from a mobile device to another mobile device, it goes through several procedures being delivered. There are two types of pathways for the SMS transmission between different mobile subscribers. Internal exchange, when two mobiles subscribers intend to exchange the SMS and both of them belong to one mobile operator company. External exchange, when two mobile subscribers intend to exchange the SMS and both belong to different mobile operators, Fig. 2 and Fig. 3 illustrates

exchanging the SMS content between two mobile subscribers [8].

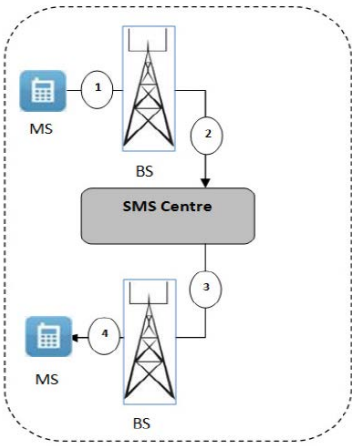

Fig. 2. Internal Exchange

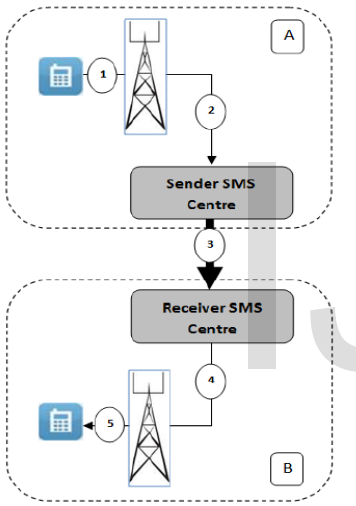

Fig. 3. External Exchange

2.2 SMS security threats

It is important that both the mobile applications and the mobile operators apply some reliable protective techniques to avoid such threats.

- SMS disclosure

Although SMS are encrypted when it is sent across the air, the encryption algorithm chosen for SMS encryption must be the network specific algorithms, such as A5 for GSM. These algorithms have been susceptible to cryptanalysis. Additionally, SMS messages are stored as plain text in the SMSC before they are successfully delivered to the intended recipient, these messages could be viewed or modified by employees in the SMSC who have access to the messaging system [9].

- SMS spoofing

SMS spoofing is injecting SMS by an attacker into the messaging networking system with an originator ID that seems to be true [10].

- SMS viruses

If a virus was attached to an SMS, it may cause an infection to the mobile phone device, especially with the rise of the technology of the modern mobile devices makes it even more susceptible to this attack [9].

3 The Proposed Model for SMS Security

The proposed model is a hybrid scheme which consists of Hash Message Authentication Code (HMAC) with Secure Hash Algorithm (SHA-256) and a cryptographic function in the form of asymmetric algorithm (RSA-2048), to secure the end-to-end SMS communications. The HMAC used to authenticate a message and to provide integrity and authenticity assurances on the message, and most importantly the HMAC will increase the RSA strength by applying randomness. The RSA fulfils the mobile security requirements. A random key K is generated by SHA-256 hashing functions which is used in the calculation of HMAC, it is known to the sender and the receiver. The randomness makes the key protected and hard to envisage by a forgery and apply strength and security making the proposed model strong. RSA is a cryptographic algorithm which is used to secure the message. RSA encrypts the originated message along with its digest; the Message Digest (MD) has been produced using HMAC SHA-256.

A modification was made in order to expand the size of the message body; this modification is concerned with the SHA-256 bit key of 32 byte. This random key will be partitioned into two parts, a static part of 28-byte that will be saved within the application and a dynamic part of 4-byte that will be sent along with the message in order to calculate the MD at the receiving side. Fig. 4 shows how the system operates at the sending side and fig. 5 shows the system operates at the receiving side.

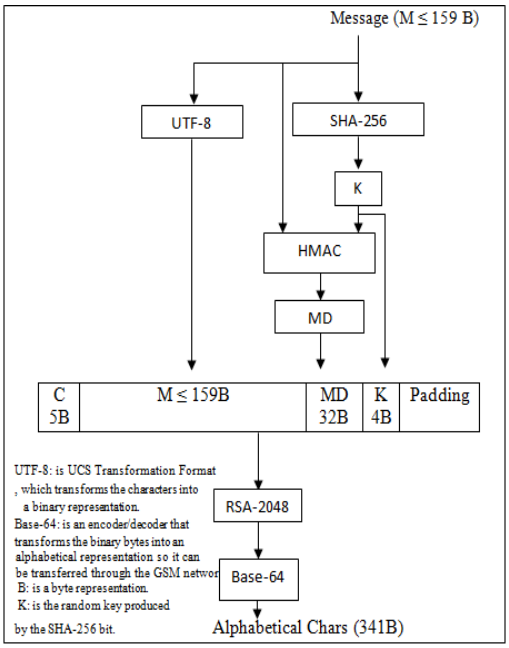

Fig. 4. Sending Side

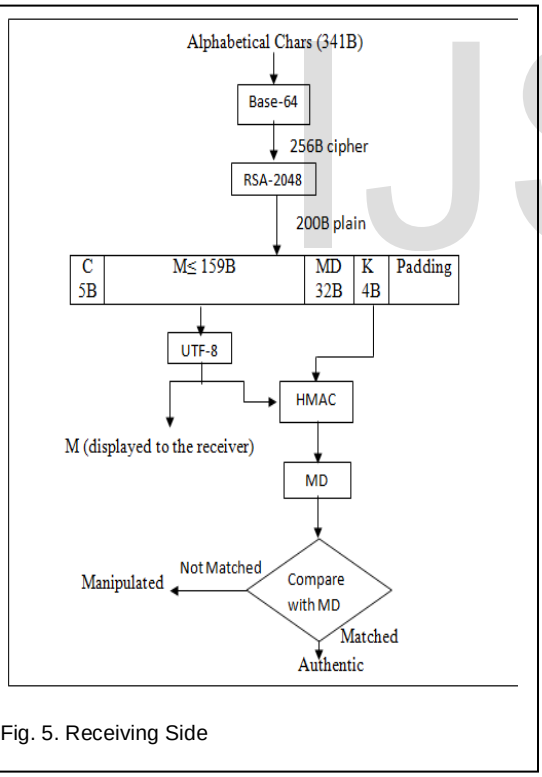

Fig. 5. Receiving Side

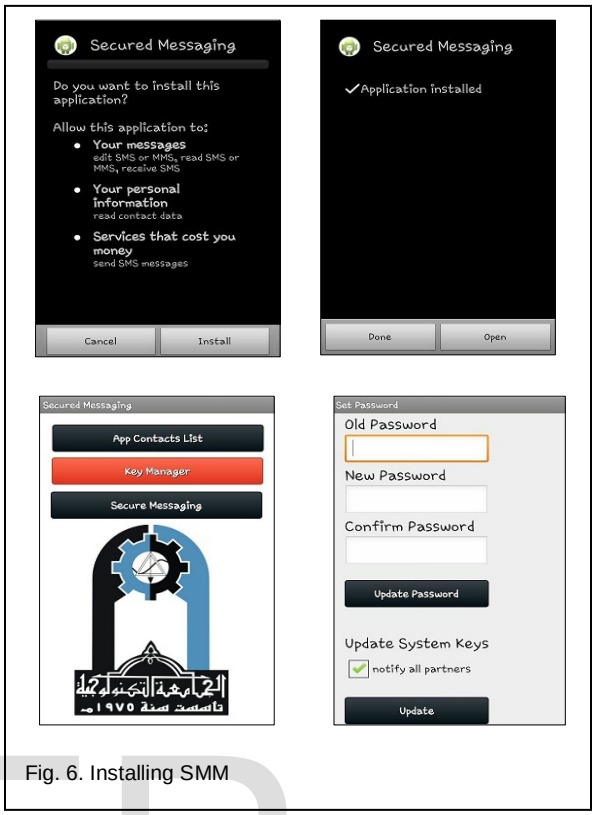

Fig. 6. Installing SMM

4 The Implementation of Proposed Model

As an implementation environment to the proposed model the Android IDE was chosen, the Eclipse juno is the program that supports android applications and it is used to build the Secure Messaging Model (SMM). The reason to choose the android platform is that it is one of the most popular and widely used platforms in the modern mobile devices. The SMM has been implemented on a real mobile device, which is the Samsung Galaxy SII, SIII, SIV, Fig. 6 shows some screenshots of the SMM, after sending, receiving and saving the PK both parties can exchange messages securely. What is needed to use this SMM is: an android mobile device and an .apk file that must be downloaded and installed so it can be runnable on the mobile device, the .apk file can be sent through the internet via FTP; or the mobile device can be connected to the pc via the USB cable with the existence of the Samsung Kies program (to define the mobile device as a runnable device).

The code of generating RSA- 2048 key pair is show in Fig. 7, Fig. 8 shows by code how the HMAC random key is generated.

The developed SMM is responsible for providing the following services without depending on third party:

International Journal of Scientific & Engineering Research, Volume 5, Issue 4, April-2014 93
ISSN 2229-5518

1. Generate the system cryptographic keys for users. This step performs the complex calculations to generate the private and public keys, it is considered a fair tradeoff that the developers has to make given the high level of security provided by the RSA key length of 2048 bit. Also the modern mobile phones have no trouble handling those computations now days with the bonus of not affecting the mobile phone performance.

2. Distribute the PK. This is done by sending the PK after its generation in a message within the SMM and the receiver has the option to save the PK within his mobile's SMM. After exchanging the PKs the two parties can communicate in a secure manner. For the first time contact, a user can't authenticate a sender, because they don't have the sender PK. Therefore, the difficulty is how a user can authenticate a sender without his PK. In this situation the SMM follows some techniques to exchange the cryptographic keys for first time as follows:

- Before the user starts any connection must add the other party's information to the SMM contacts list, such as name and phone number. Therefore, any request from any party that does not exist in the SMM contacts list will be ignored immediately.
- A fake message containing a fake PK will be diagnosed immediately as "a fake PK message" without the Save option.

Those are very useful to protect the valid users from invalid user while increasing the level of the security as well.

3. Save the public keys sent by users who wants to communicate. The SMM is responsible for storing the received public keys from senders that want to communicate in a secure manner; it uses the SQ Light Database to store the PKs within the application along with other information. The size of the public key for the user does not exceed 2048 bit. Most of today's mobile phones are designed with an extra memory, so users can easily store the public keys on their phones.

The SMS encryption code is shown in Fig. 9; Fig. 10 shows how the SMS decryption code and Fig. 11 shows the code that compares the received MD with the calculated MD (for integrity and authenticity purposes).

```
public void onClick2(View v) throws NoSuchAlgorithmException{
    shared_preferences = getSharedPreferences("MyKeysSP",
            MODE_PRIVATE);

        shared_preferences_editor = shared_preferences.edit();


        long startTime = System.currentTimeMillis();
    KeyPairGenerator kpg = KeyPairGenerator.getInstance("RSA");
       kpg.initialize(2048);
       KeyPair kp = kpg.genKeyPair();
       PublicKey publicKey = kp.getPublic();
       PrivateKey privateKey = kp.getPrivate();

       byte[] PUKBytes = publicKey.getEncoded();
       byte[] PRKBytes = privateKey.getEncoded();

       String Str_PUK= Base64.encodeToString(PUKBytes ,Base64.URL_SAFE);
       String Str_PRK= Base64.encodeToString(PRKBytes ,Base64.URL_SAFE);
```

Fig. 7. RSA Keys Generation

```
private void appendDigest() throws InvalidKeyException, NoSuchAlgorithmException{


    byte[] bytRand = new byte[32-HMACSSKeyL];
    new Random().nextBytes(bytRand);

    byte[] SSKey=HMACSSKey.getBytes();


    byte[] Digest=subArray(0,HMACSSKeyL-1 ,SSKey);
    Digest=append(Digest,bytRand);


  this.input=append(this.input,calcHmac(this.input,Digest));



  this.input=append(this.input,bytRand);

}
```

Fig. 8. HMAC Random Key Generation

```
public void SendSMS(View view) throws InvalidKeyException, NoSuchAlgorithmException,
NoSuchPaddingException, IllegalBlockSizeException, BadPaddingException, UnsupportedEncodingException,
InvalidKeySpecException{

    Uri uri = Uri.parse("smsto:" + ContactNumber);
    Intent intent2 = new Intent(Intent.ACTION_SENDTO, uri);
    String Plaintext=editBox.getText().toString();
    String Ciphertext;
    byte[] Plaintext_byt=Plaintext.getBytes("UTF-8");

    if(Plaintext.length() >0)
    {

        if(Plaintext_byt.length < PTLength)
        {
            String Str_PUK = ContactPK;
            byte[] Byt_PUK=Base64.decode(Str_PUK, Base64.URL_SAFE);
    PublicKey PUKK= KeyFactory.getInstance("RSA").generatePublic(new X509EncodedKeySpec(Byt_PUK));

    EDSystem mysys=new EDSystem(PUKK, null, true,Plaintext_byt);

    byte[] Enc_Byt=mysys.blockCipher();
    String Enc_Str=Base64.encodeToString(Enc_Byt ,Base64.URL_SAFE);
        intent2.putExtra("sms_body",Enc_Str);
        startActivity(intent2);

        }
```

Fig. 9. Encrypt SMS

```
PrivateKey PRKK = KeyFactory.getInstance("RSA").generatePrivate(new PKCS8EncodedKeySpec(Byt_PRK));

EDSystem mysys=new EDSystem(null, PRKK, false,Base64.decode(SMString2, Base64.URL_SAFE));
byte[] Dec_Byt=mysys.blockCipher();
String Dec_Str=new String(Dec_Byt,"UTF-8");

TextViewSMS.setText(Dec_Str);
TextViewSMSTitle.setText("Plaintext");

}
```

Fig. 10. Decrypt SMS

```
private void isAuth(byte[] input) throws InvalidKeyException, NoSuchAlgorithmException{

    byte[] Digest=new byte[DgtLength];
    byte[] MsgDigest=new byte[DgtLength];
    byte[] DKey=new byte[DgtKLength];
    byte[] Msg=new byte[input.length-(DgtLength+DgtKLength)];


    Msg=subArray(0, input.length-1-(DgtLength+DgtKLength), input);


    MsgDigest=subArray(input.length-(DgtLength+DgtKLength),input.length-1-DgtKLength, input);
    DKey=subArray(input.length-DgtKLength,input.length-1,input);
    DKey=append(subArray(0,HMACSSKeyL-1 ,HMACSSKey.getBytes()),DKey);


    Digest=calcHmac(Msg,DKey);
    this.isAuth= Arrays.equals(Digest,MsgDigest);

}
```

Fig. 11. Comparing MD's

The system must be tested to measure its performance, strengths and weaknesses. Three mobile devices were used to implement and test the SMM, Table 1 illustrate the results of encryption, decryption operations and Table 2 illustrate the MD computation time.

Comparing the key generation and the testing results with the NTRU algorithm [11] that is considered secure and strong algorithm, it is noted that implementing the RSA-2048 [12] is better in the timing process, both test of English text as shown in Table 3.

Table 1
Testing Results

Device	Key Generation	Input Language	Encryption	Decryption
SIII	2.500 ms	Arabic	9.42 ms	37.42 ms
		English	8 ms	36.44 ms
SIV	1.787 ms	Arabic	3.62 ms	28.8 ms
		English	1.08 ms	26.9 ms

Table 2
Message Digest Computation

Device	Key Generation	Input Language	Message Digest
SIII	2.500 ms	Arabic	1.64 ms
		English	1.34 ms
SIV	1.787 ms	Arabic	0.34 ms
		English	0.5 ms

5

Table 3
Comparative Results

Algorithm	Key Generation	Encryption	Decryption
RSA-2048	1787 ms	108 ms	269 ms
NTRU-251	9617 ms	515 ms	1132 ms

5 Discussion

The proposed model depends on a hybrid of two algorithms. First is the HMAC-SHA256. The security of the message authentication mechanism presented here depends on cryptographic properties of the hash function H: the resistance to collision finding (limited to the case where the initial value is private and random, and the output of H function is not explicitly available to the attacker), the inner computations and inner hash algorithms used by the HMAC are unknown to attackers and most importantly the MD calculated by HMAC-SHA256 is encrypted with RSA, so it is extremely difficult and time consuming operation trying to crack it down.

Second, the cryptographic RSA 2048 function is discussed; it is one of the strongest algorithms, providing a high security level for the sensitive information being sent along in the form of encrypted SMS. Trying to factorize RSA 2048 will be infeasible and impossible to perform, duo to its proof by NIST that RSA 2048 is still standing against the factorization attack.

Exchanging encrypted messages stage is the permanent stage that comes after the key exchange/update finished successfully, the users will be able to use the public keys to encrypt any new messages for the current keys. And will also be able to find out the message authenticity by the use of HMAC-SHA256 to calculate and compare MD. As a result, users will be able to verify the integrity of the sender's message. Any incoming message by anyone not included in the secure application contacts list will be ignored. Any fake PK message will be recognized immediately by the SMM as a "fake PK message" this is done by taking advantage of the Base-64 and comparing its character restriction order (A-Z, a-z, 0-9, - and _) with the upcoming PK messages.

6 Conclusions

The proposed model is a hybrid cryptographic scheme of RSA-2048 and HMAC-SHA256 algorithms to provide the protection for end-to-end SMS security weaknesses. The proposed model guarantees the requirements of security services. It is implementable completely on the user's mobile phones independently from any third party, without any additional hardware. Android has been selected to be the development environment due to its

International Journal of Scientific & Engineering Research, Volume 5, Issue 4, April-2014 95
ISSN 2229-5518

popularity and distinguishing features, working with Eclipse and the integrated ASDK along with JDK, all of them provide the support for implementing the application's cryptographic algorithms. The SMM is running well and it executes all the required operations. In term of speed, the SMM runs with minimal delay and it was able to complete all of cryptographic operations in less than a second. The SMM runs fast enough so it has no slowdown in the mobile device's performance.

REFERENCES

[1] Yu Loon Ng, "SHORT MESSAGE SERVICE (SMS) SECURITY SOLUTION FOR MOBILE DEVICES", December 2006.

[2] Johnny Li-Chang Lo, Judith Bishop, J.H.P Eloff, "SMSSec: an end-to-end protocol for secure SMS", Computer Science Department, University of Pretoria, South Africa, 2008.

[3] STALLINGS,"Cryptography and Network Security Principles and Practices", Prentice Hall, W. 2005.

[4] "Mobile Commerce: opportunities and challenges - A GS1 Mobile Com" White Paper February,2008 Edition

[5] Tarek M Mahmoud, Bahgat A. Abdel-latef, Awny A. Ahmed & Ahmed M Mahfouz, "Hybrid Compression Encryption Technique for Securing SMS", International Journal of Computer Science and Security (IJCSS), Volume (3): Issue (6) P473-P481

[6] Anita & Nupur Prakash, "Performance Analysis of Mobile Security Protocols: Encryption and Authentication", International Journal of Security, Volume (1) : Issue (1), June 2007

[7] J. Li-Chang Lo, J. Bishop and J. Eloff. "SMSSec: an end-to-end protocol for secure SMS", Computers & Security, 27(5-6):154-167, 2007

[8] Medani1*, A. Gani1, O. Zakaria2, A. A. Zaidan3,4 and B. B. Zaidan, "Review of mobile short message service security issues and techniques towards the solution", 2011.

[9] "Short Message Service Security", The Government of the Hong Kong Special Administrative Region, 2008.

[10] He Rongyu, Zhao Guolei, Chang Chaowen, Xie Hui, Qin Xi and Qin Zheng, "A PK-SIM card based end-to-end security framework for SMS", Computer Standards & Interfaces, 2009.

[11] Sameer Hasan Al-bakri* and M. L. Mat Kiah, "A novel peer-to-peer SMS security solution using a hybrid technique of NTRU and AES-Rijndael", Department of Computer System and Technology, Faculty of Computer Science and Information Technology, University of Malaya, 50603 Kuala Lumpur, Malaysia., 2010.

[12] Elaine Barker, William Barker, William Burr, William Polk, and Miles Smid, "C O M P U T E R S E C U R I T Y", NIST Special Publication 800-57, 2012.