Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 4, Issue. 6, June 2015, pg.1057 – 1064

RESEARCH ARTICLE

Secure SMS Communication in Android based System with Two Stage Protection

Assistant Prof. Subhasis Banerjee[1], Prof. Utpal Roy[2]

[1]Department Computer & System Sciences, Visva-Bharati, India

[2]Department Computer & System Sciences, Visva-Bharati, India

[1] bsubh@yahoo.com; [2] roy.utpal@gmail.com

Abstract— In this paper, we present a scheme for secure SMS communication. The suggested scheme is based on private key crypto-system. This secure SMS will be suitable to work on android platform. The Android platform has been dealt as a topic of mobile security because Android is an open platform whose sources can be observed by anyone. The malicious code inserted in any application can perform to intercept and forward a SMS with hiding the action. In order to protect the SMS on the platform, the designed scheme provide confidentiality of a SMS and the integrity. In this paper, we present Common private Key Cryptography for SMS security choosing the key from a table randomly. We have also used simple hashing technique to keep the integrity of the message intact.

Keywords— BTS, UE, SMSC, PKI

I. INTRODUCTION

A. SHORT MESSAGE SERVICE (SMS)

SMS stands for short message service. It is a method of communication that sends text between cell phones, or from a PC or hand-held to a cell phone. The maximum size of the text messages is: 160 characters (letters, numbers or symbols in the Latin alphabet). For other alphabets, such as Chinese, the maximum SMS size is 70 characters

B. Need for secure data transmission

Information security means [1] protecting information. It secures information systems from unauthorized access, use, disclosure, disruption, modification, perusal, inspection, recording or destruction. Maintaining privacy in our personal communication is something everyone desires. Encryption is a means to achieve that privacy. It was invented for the very same purpose. As short message service (SMS) is now widely used as a business tool; its security has become a major concern for business organization and customers. So there is a need for an end to end SMS encryption in order to provide a secure medium for communication.

C. **Working of SMS**

It is well known that SMS service is a cell phone feature but indeed, SMS can also work on other computing devices now a days , such as PC, Laptop, or Tablet PC as long as they can accept SIM Card. SIM Card is needed because SMS service needs SMS centre client which is built in on the SIM Card.

i) BTS

A base transceiver station (BTS) is a piece of equipment that facilitates wireless communication between user equipment (UE) and a network. UEs are devices like mobile phones (handsets), WLL phones, computers with wireless internet connectivity, WiFi and WiMAX devices and others.

ii) MSC

The mobile switching centre (MSC) is the primary service delivery node for GSM/CDMA, responsible for routing voice calls and SMS as well as other services (such as conference calls, FAX and circuit switched data). The MSC sets up and releases connection. This end to end connection handles mobility and hand over requirements during the call and takes care of real time prepaid account monitoring etc..

iii) SMSC

When SMS is transmitted from a cell phone, the message will be received by mobile carrier's SMS Centre (SMSC), [2] which does destination finding, and then send it to destination devices (cell phone). SMSC is SMS service centre which is installed on mobile carrier core networks. Beside as SMS forwarding, SMSC also acts as temporary storage for SMS messages. So, if the destination cell phone is not active, SMSC will store the message and then deliver it after the destination cell phone is active. As additional, SMSC also notify the sender whether the SMS delivering is success or not. However SMSC cannot store the SMS message forever since the storage capacity is not unlimited. During the SMS delivering, sender cell phone and SMSC is actively communicating. So, if the non active destination cell phones become active, SMSC directly notifies the sender cell phone and tell that the SMS delivering is success. This is how the SMS works in general.

So we can say that after the SMS message is sent by the user, SMS Centre (SMSC) is used to store the SMS messages in order to forward them to the target mobile device. SMSC uses Store-and-forward technique to store messages in order to forward to the target device. If the (Home Location Register) HLR of target mobile device is active, then SMSC will transfer the SMS message to target mobile device. SMSC will receive the verification message that confirms the delivery of SMS message to target device . Unencrypted SMS messages are stored in SMSC; therefore, SMSC staff can view and modify the content of SMS message. Many SMSCs can also keep the copy of SMS message for billing and auditing purposes. Therefore, it becomes easy for attackers to view SMS messages through SMSC . After attacking SMSC, attacker can read the SMS messages. Several Cryptography methods [3] have been used to reduce the SMS security threats and provide enough security to mobile devices. But these encryption techniques can't perform their activity in a complete manner since it affects the performance of mobile devices in terms of power and battery life constraints. Symmetric Cryptography is the type of encryption used to provide end-to-end security to SMS messages. It is also good for mobile devices due to their limited resources, i.e., limited power/energy, in-sufficient memory and less processing power. It uses the shared secret key between two parties in order to protect SMS message communication. Key distribution mechanism remains in-secure, since if an attacker intercepts the key distribution process and intercepting the key, he/she can easily modify the SMS message contents. Therefore, Key distribution is quite difficult and insecure in symmetric key cryptography. DES and AES are the examples of symmetric key cryptography. The key distribution problem is solved by Asymmetric cryptography [4] by using pair of keys (i.e. private and public) for communication. Sender is using public key for communication while private key is used in order to decrypt the message. Man-in-the-middle attack is common in public key cryptography. Public key infrastructure (PKI) is then used to improve the deficiency of public key cryptography. Although Asymmetric encryption is strong and key distribution is also very easy in it, but, it is avoided because of its computational overhead.

Nevertheless, mobile devices have improved their memory capacity as well as their performance. Energy efficiency and battery technology is also improved in order to extend the operational time of mobile devices. Besides of these developments, [5] it is still a research question that whether symmetric and asymmetric encryption can fully provide their advantages to secure mobile SMS messages.

II. LITERATURE SURVEY

Mohsen Toorani et al., [6] provided the introduction of new Secure SMS Messaging Protocol (SSMS) for the Mobile Payment systems. It being an application layer protocol is intended for GSM users as a secure bearer in the mobile payment systems. It uses elliptic curve based public key solution which uses public key as secret key for symmetric encryption.
Marko Hassinen provided application solution named "Safe SMS", using java for achieving confidentiality, integrity and authentication in SMS without any additional hardware for ensuring message is not tempered and authenticates sender. Safe SMS has two methods for encrypting via Quasigroup and Blowfish. S. H. Shah Newaz et al., proposed scheme for the enhancement of SMS security system for GSM users. Thereby, incorporating digital signature over cipher which is converted so by existing encryption schemes is made compatible to GSM security infrastructure. Encryption can be done with the existing GSM encryption algorithm, called A8. Then the encrypted message will create hash and finally it will be digitally signed. Thus, signed encrypted message will be transmitted.

Mary Agoyi et al., evaluated encryption and decryption time for three algorithms RSA, Elliptic curve and ElGamal to which plain text of different sizes is provided based on results one is chosen for further encryption.
Their performance evaluation in securing SMS shows that key generation, encryption and decryption time increases with an increase in key size. Large key size algorithms are not suitable for SMS encryption due to small memory and low computational power of mobile phones. Na Qi Jing Pan Qun Ding did improvements on RSA algorithm because the SMSC will filter out the characters which are out of prescribed limit, thus the cipher text can't reach the destination. Thus, they also used FPGA based on high speed processing tools to implement the RSA algorithms and apply it in mobile phone short message encryption system. Ch. Rupa et al., proposed accost effective scheme which uses a concept called Cheating Text. The original message is embedded in a meaningful text called cheating text. Here, index table called (Real Message Index File) RIF file is hashed and sent to the receiver along with the cheating text in which the original message is embedded. Authentication is achieved by verifying the hash value of the plain text.

III. Android architecture and security related issues

The Android platform is the most popular platform for mobile devices. It is designed in a way, that applications can be easily installed through online application markets. In order to secure such a system, Android implements a security concept [7] [8]which protects the system as well as the applications by isolating the application context and permission Management. Android is not a single piece of hardware; it's a complete, end-to-end software platform that can be adapted to work on any number of hardware configurations. Everything is there, from the boot loader all the way up to the Applications. In the following subsection we discuss the Android architecture.

A. Architecture

The architecture of Android is based on a Linux Kernel and adopts many concepts, which are common on Unix systems. Android uses different system users and applies file system permissions which isolate applications. In order to allow communication and interaction beyond this isolation, Android provides interfaces, which are protected by a permission management.

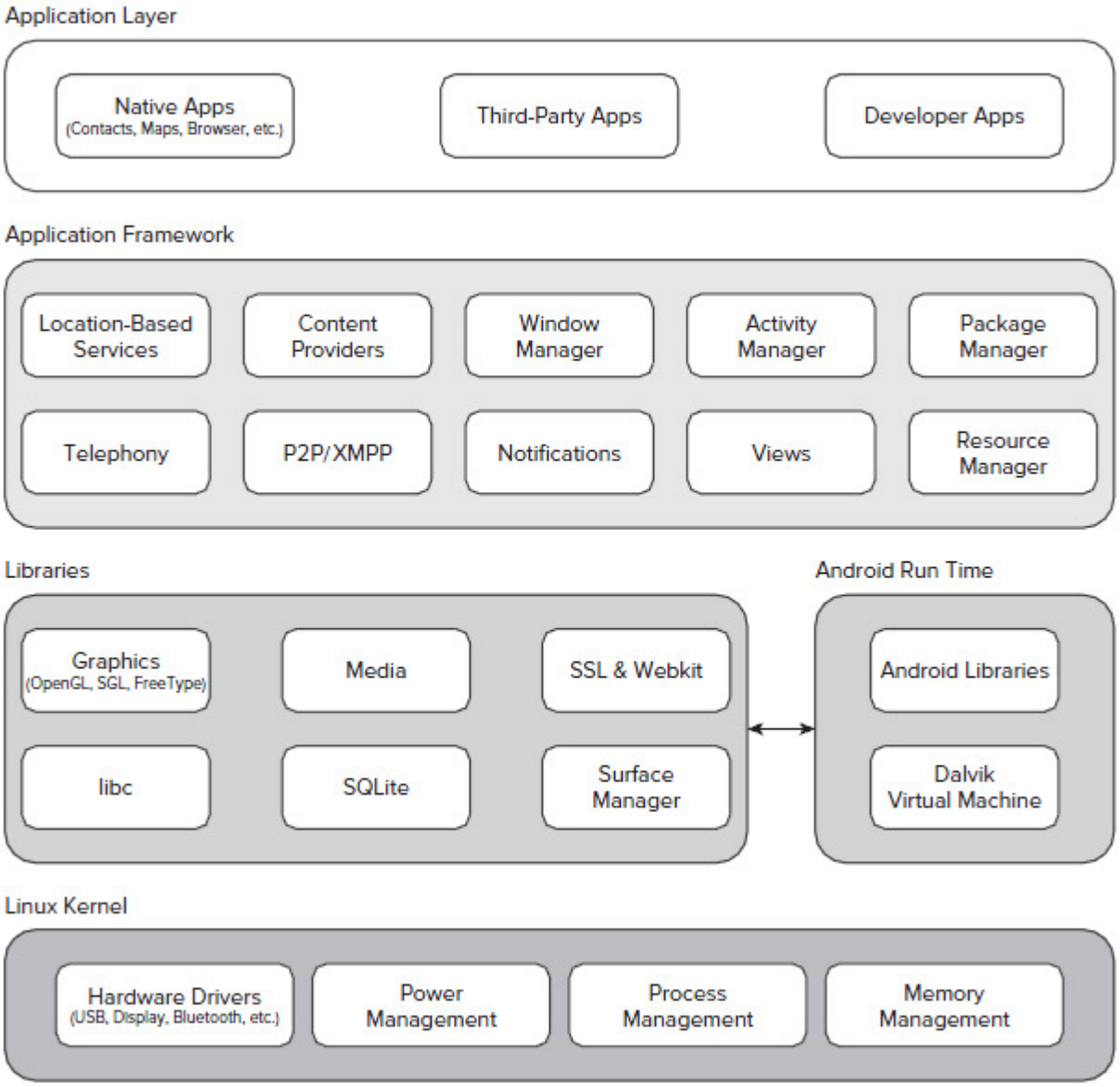

Fig-1

The Android architecture consists of five main component groups, as shown in figure- 1. The underlying Linux kernel manages any hardware access and provides interfaces for device specific drivers, which were provided by hardware vendors. The configuration of the kernel on Android has been adjusted specially for mobile devices. Therefore, many features, which are enabled on desktop and server systems, have been disabled in order to reduce memory and energy consumption. The second component group contains native libraries, which provide different services e.g. for graphic rendering, cryptographic functions and database access etc.
These components are executed natively and can be used within Android applications by loading them into their process context. So, these components are stored as shared objects. Such a process of an application also contains a copy of the Dalvik Virtual Machine (DVM), as part of the third component group shown in figure 1. The DVM executes Android applications, which are delivered in form of Dalvik bytecode. Within the DVM, an application can make use of services, which are provided by the Android Application Framework and native libraries. In this way the Android Application Framework can ensure compatibility between different systems and also restrict data access by permission checking.

B. Security

Android is a multi-process system, in which [9]each application (and parts of the system) runs in its own process. Most security between applications and the system is enforced at the process level through standard Linux facilities, such as user and group IDs that are assigned to applications.

C. SMS Applications with Android

Now a days SMS is one of the most used application on mobile phones. It is very useful for small person to person message communications. It can also be used in a variety of ways [10][11]within applications beyond it's normal use. For instance, it can be used to exchange small amounts of data between phones running the same application. With most built-in functionality in Android we can use SMS message, utilizing the normal SMS Activity. We can also develop an application which receives SMS messages. In this type situation we need to tell Android that we want to handle incoming SMS messages. SMS handling in Android is managed by launching a specific class when an SMS comes in. Android Application Framework can restrict data access by permission checking. But users are not offered any fine-tuned control of what permissions to grant the applications. If an application requests for access or permissions the user are not comfortable with granting, so the only choice is to not install the application. There are suggestions of improvement to the permission system, such as the ability to grant only a subset of requested permissions, and also advocate for the inclusion of Security Enhanced Linux (SELinux) to further protect the Linux kernel using Mandatory Access Control policies.

Access control is a very useful measure against many types of attacks. The rapid increase of malware on Android can compromise a device and steal sensitive information. Most of these malware comes in the form of Trojans which provide a method to inject arbitrary code into application without invalidating their signature. So integrity of any data or message must be kept intact to save the system from Trojans. There was a larger security analysis of Android carried out by Berger et al. They found discrepancies between the documentation of Bluetooth communication available to developers and the actual implementation in the Android system itself. Several weaknesses are identified regarding communication, both between applications and between an app and the Internet [12][13]. Since Google does not use a manual review process it is important that the tools used to automatically review submitted applications can have weak configurations. This type of drawbacks will make Android based communication channels vulnerable.

So in this situation we must ensure the SMS communication to be secure. We have to keep in mind that SMS is used not only for personal communication now a days but it is also used for secure system messages, secure e-commerce communications etc.

IV. **Working procedure of our secure SMS application**

Our encryption procedure is combination of Affine cipher and Hill cipher.

In Affine cipher encryption and decryption are as follows :

Encryption procedure : $e(x) = ax + b \bmod 26$

Decryption procedure : $d(y) = a^{-1} (y - b) \bmod 26$

Where gcd (a,26) = 1, a, b are positive integers less than 26.

Here x and y are plaintext and cipher text respectively.

The Hill cipher is developed by the mathematician Lester Hill.

Now a days some modified version of Hill cipher is used for encryption in practical cases[14]. The encryption of old Hill Cipher are as follows.

HILL CIPHER

The core of Hill-cipher is matrix manipulations. It is a multi-letter cipher, developed by the mathematician Lester Hill in 1929.

For encryption, algorithm takes m successive plain text letters and instead of that substitutes m cipher letters. In Hill cipher each character is assigned a numerical value like:

a=0,

b=1,

…..

…..

z=25.

The substitution of cipher text letters in place of plain text leads to m linear equations. For m=3, the system can be described as follows:

C1=(K11P1+K12P2+K13P3)MOD26
C2=(K21P1+K22P2+K23P3)MOD26
C3=(K31P1+K32P2+K33P3)MOD26

This can be expressed in terms of column vectors and matrices as follows :

C=KP

Where C and P are column vectors of length 3, representing the plain text and the cipher text and K is a 3*3 matrix, which is the encryption key. All operations are performed mod 26 here. Decryption requires the inverse of matrix K. The inverse K^{-1} of a matrix K is defined by the equation .K K^{-1}= I where I is the Identity matrix. The inverse of a matrix doesn't always exist, but when it does it satisfies the proceeding equation.

K^{-1} is applied to the cipher text, and then the plain text is recovered. In general terms we can write as follows:

For encryption: C=Ek(P)=PK

For decryption: P=Dk(C) =CK^{-1} = PKK^{-1}=P

Here the encryption is multiplication by a key matrix and decryption is by multiplication of the inverse of the key matrix. For our purpose we have taken the key matrix of order 3 x 3 to reduce the computational overhead.

Encryption

Suppose the key is :

$$K = \begin{bmatrix} 11 & 8 \\ 3 & 7 \end{bmatrix}$$

So K^{-1} $= \begin{bmatrix} 7 & 18 \\ 23 & 11 \end{bmatrix}$ in modulo 26

Suppose we want to encrypt the plaintext july. So the vector format of our plaintext is

[9,20] corresponding to ju and

[11,24] corresponding to ly.

We compute the encrypted text as follows :

$$[9\ \ 20] \begin{bmatrix} 11 & 8 \\ 3 & 7 \end{bmatrix} = [99+60 \quad 72+140] = [3 \quad 4] \text{ in modulo 26}$$

$$\text{And } [11\ \ 24] \begin{bmatrix} 11 & 8 \\ 3 & 7 \end{bmatrix} = [121+72 \quad 88+168\ ] = [11\ \ 22] \text{ in modulo 26}$$

Hence ,the encryption of july is DELW.

Similarly we can decrypt the encrypted text as follows :

$$[3\ \ 4] \begin{bmatrix} 7 & 18 \\ 23 & 11 \end{bmatrix} = [9\ \ 20]$$

$$[11\ 22] \begin{bmatrix} 7 & 18 \\ 23 & 11 \end{bmatrix} = [11\ \ 24]$$

Hence we have got the correct plaintext.

Key Matrix generation procedure:

1. Generate a n x n random matrix with positive integers as elements.
2. Reduce the matrix in modulo 26 and it is the key matrix
3. Calculate its inverse in modulo 26
4. If the matrix is not inevitable, go to Step 1
5. Store the key matrix and its inverse

Our application works in the following way:

1. The sender opens the application which is basically a GUI based editor.
2. Sender type the message on the above editor and at the end of the message one message digest is added using a hash value. We can check the integrity of the message using it.
3. The message and the added hash value is encrypted using Affine cipher.
4. User chooses the key matrix from a list, maintained by the software for the encryption algorithm
5. User chooses the key randomly from a key spool and it is reflected by a number. This number is communicated to sending site.
6. Then the number and the encrypted message is communicated to the other site
7. With the help of the communicated number receiver first finds the key from the spool of the same software.
8. The message is decrypted using the inverse of the key matrix.
9. The output message is then decrypted by Affine cipher.
10. The message integrity is checked by matching the received message digest calculated in the receiving site.
11. If the message size crosses the size limit of the SMS it will be fragmented in the sending site and re-joined in the receiving site. The fragmentation is identified by special bit pattern added at the front of the text.

Security Analysis: The proposed crypto-system provides strong security against the known plain text attack since n equations cannot be used for solving an unknown n×n matrix and n^2 unknown parameters. Furthermore Affine cipher is used as a second level of protection. We have added a hashed message digest. This ensures that the integrity of the message is intact.

V. CONCLUSIONS

The security of the SMS in mobile communication is a big as well as challenging issues in modern days .SMS services are inevitably in mobile banking, e-commerce, defence applications and in many others. Computational complexity of most of the well known public key crypto system is very high. So the incorporation of public key crypto-system in case of SMS security is not advisable in a low power driven Android based system. In our proposed scheme we have chosen a private key crypto system which is cost effective considering the computational complexity. Furthermore we check the integrity of the message by the use of hash function. Our random selection of key from a key spool is very good technique to increase the overall security level of the system.

REFERENCES

[1] Information Security, http://www.infosec.gov.hk/english/technical/ files/ short.pdf.
[2] Micro System Center, http://technet.microsoft.com/enus/library/ cc181234.asp
[3] W. Stallings, "Cryptography and network security", Prentice Hall, 2006, New Jersey, United State
[4] Mary Agoyi, Devrim Seral, "SMS Security: An Asymmetric Encryption Approach", Sixth International Conference on Wireless and Mobile Communications, 2010@IEEE, pp 448-452.
[5] Jhonny Li-Chang Lo,Judith Bishop, and J.H.P Eloff, "SMSSecan end to end protocol for secure SMS,"comput.security,vol.27,2008.
[6] M. Toorani and A. A. Beheshti Shirzai, SSMS . A Secure SMS Messaging Protocol for the M-Payment Systems, IEEE Symposium on Computers and Communications, 2012, 700-705
[7] Adrienne Porter Felt, "Android Permissions: User Attention, Comprehension, and Behavior", Symposium on Usable Privacy and Security (SOUPS) 2012, July 11-13,2012, Washington, DC, USA
[8] http://source.android.com/posts/opensource

[9] Manisha Madhwani, "Cryptography on Android Message Application Using Look Up Table And Dynamic Key (Cama)", IOSR Journal of Computer Engineering (IOSRJCE) ISSN: 2278-0661, ISBN:2278-8727 Volume 6, Issue 2 (Sep-Oct. 2012), PP 54

[10] J .P.Albuja and E.V.Carrera, "Trusted SMS communication on mobile devices",11th Brazillian workshop on Real Time and Embeded system, pernambuco,Brazil,2009.

[11] A. Singh,S.Maheshwari,S.Verma and R. Dekar, "Peer to Peer Secure Communication in Mobile Environment: A Novel Approach",International Journal of Computer Application Vol 52,2012.

[12] Artemios G. Voyiatzis, "Increasing Lifetime of Cryptographic Keys on Smartphone Platforms with the Controlled Randomness Protocol".

[13] Digital Signature, http://en.wikipedia.org/wiki/Digital_signature.

[14] Liam Keliher, Anthony Z. Delaney, "Cryptanalysis of the Toorani-Falahati Hill Ciphers", IEEE, pp 436-440, 2013.

