Journal of Information Assurance and Security.
ISSN 1554-1010 Volume 6 (2011) pp. 369–378


# A Novel Model for Secure Mobile SMS Banking

**Farzad Tavakkoli[1], Reza Ebrahimi Atani[1,2] and Shahriar Mohammadi[3]**

[1]The University of Guilan (Anzali International Branch), Department of Information Technology
Anzali Trade-Industrial Free Zone Organization, Anzali, Iran
*f.tavakkoli@msc.guilan.ac.ir*

[2] The University of Guilan, Department of Computer Engineering,
P.O. Box 3756, Rasht, Iran
*rebrahimi@guilan.ac.ir*

[3] Khajeh-Nasir Toosi University of Technology, Department of Industrial Engineering,
Information Technology Engineering Group, Tehran, Iran
*smohammadi40@yahoo.com*

*Abstract*: **Modern banking systems allow customers to conduct financial transactions on new communication channels. SMS banking is a technology-enabled service offering from banks to its customers, permitting them to operate selected banking services over their mobile phones using SMS. However, the security of mobile SMS payment and banking has topped the list of concerns for most of the customers. The aim of this work is to investigate the security loopholes in SMS banking and propose a novel end-to-end encryption system to make mobile SMS banking secure.**
***Keywords***: SMS Banking, One Time Password, Stream Ciphers, Hash Functions, End-to-End Encryption, Mobile Banking

## I. Introduction

Today, pervasive use of wireless networks and mobile devices changed our lifestyle's schema seriously. Among mobile devices, the mobile phone usage is very important because of the high penetration rate. So it encourages the industry owners and service providers to present almost any internet-based services on mobile phones. One of the mobile services which have contributed a lot in the growing of E-Commerce is M-Commerce. On the other hand a large part of commerce is financial operations and transactions that are mostly done by banks and financial institutions.
The main reason that Mobile Banking scores over Internet Banking is that it enables anywhere-anytime banking. Customers now do not need to access a computer terminal to access their banks, they can now do so on the go - when they are waiting for their bus to work, when they are traveling or when they are waiting for their orders to come through in a restaurant. These activities may include account transactions, transfer funds between their accounts and others, request a cheque, announced banned bank cards and accounts and may even pay the bills. The Mobile SMS banking system is based on the exchange of SMS between customers and the bank.
A variety of communication technologies are used in mobile banking, such as WAP-based mobile banking, Bluetooth-based mobile banking and mobile banking based on SMS. Table 1 shows the advantages and disadvantages of different mobile banking services.

*Table 1*: Comparison of mobile banking Channels

| M-banking Channel | Ubiquity | Simplicity of usage | Security | Cost |
|---|---|---|---|---|
| WAP | Low | Low | High | High |
| Bluetooth | Medium | Medium | Low | Low |
| SMS | High | High | Medium | Medium |

Short Message Service, better known as SMS is a service that enables the sending of text messages over a mobile cellular network [1]. The SMS is a store-forward service, in other words, the messages can be stored in the network until they are collected by the recipient's terminal equipment (such as a mobile phone or device that can be connected to the network). This means that short messages are not sent directly from sender to recipient, but always via an SMS Center (SMSC) instead. The service center is responsible for the collection, storage, and delivery of short messages [2].
Originally, SMS design was developed as part of the GSM network between 1986-1991, but today SMS is employed even in a wide range of cellular networks such as CDMA, D-AMPS and even satellite communications networks. This service initially was designed and used to exchange regular and non-classified information, but with growing M-Commerce it became the main tool of business strategy.

Although all the advantages SMS banking brought to the community, development of new analysis techniques and also the growth of technology has risen many risks on the secure transmission of the client's private information. This is due to the non encrypted exchanged text messages. On the other hand, encryption algorithms A5 and different versions of A5 are used in GSM protocol, but these encryption

techniques are vulnerable to various types of mathematical analysis [3], [4], [5], [6], [7] and also physical attacks [8]. The worst of all, the messages are stored in the SMSC in plain text format and if the physical security of the SMSC is compromised by an external intruder or even internal staffs, another point of vulnerability in SMS security will be accrued. To overcome these weaknesses, we can use SMS encryption model between sender and receiver. This model is referred to as end-to-end encryption (E2EE). Since confidentiality and integrity over the channel between two parties are not provided by current mobile networks (such as GSM and UMTS) and therefore they have to be implemented at applications for the cell phones [9]. So with this scenario End-to-End encryption can be obtained in the application layer without any need to change the infrastructure of the current mobile networks. A sample End-to-End ecryption model for SMS Banking is shown in figure 1.

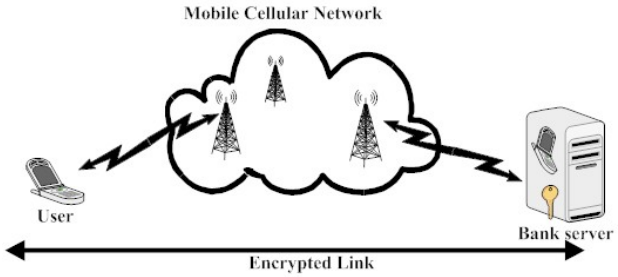


**Figure. 1**: End-to-End ecryption model for SMS Banking.

The rest of the paper is structured as follows: a general description of the previous works is presented in section II. Section III describes the novel SMS banking model. Security analysis of the proposed model is in section IV and the paper concludes in section V.

## II. Related Works

In this section we present an overview of some related works. Some researchers have raised steganography for securely exchanging secret/classified information via SMS [10], [11], [12]. Steganography is the art and science of writing hidden messages in such a way that no one, apart from the sender and intended recipient, suspects the existence of the message, a form of security through obscurity. The steganography have received a great attention in recent years by security experts. In the implementation of stegenography, the main goal is to hide information in the cover of another medium, so that others will not notice the hidden information [11].
With this method only confidentiality as one of the security principles can be obtained but it is not so robust to use in financial transactions trough SMS. On the other hand this technique needs a mutual key agreement between two ends and the key, which is called stego-key must be exchanged through in an insecure channel like SMS which is also a vulnerability in message exchange.
Cryptography is another method for achieving E2EE in SMS transactions. Many papers have discussed using symmetric/asymmetric key cryptography and have tried to approve security principles with using digital signature and hash functions [13], [14]. The common model used for securing SMS is based on symmetric cryptography [15]. The most well known Symmetric algorithm, Advance Encryption Standard (AES) is commonly used For SMS encryption. AES was designed such that an efficient software implementation is possible. It has a small footprint and therefore is a good choice for implementation in recourse constraints devices such as mobile phones. AES demands small computing power, therefore, applications can be written for the most widespread programming platform Java Platform, Micro Edition [15]. Like all symmetric key encryption algorithm, there is a need for exchange of encryption keys via a secured channel which is one of the disadvantages of this model.
The second option is to use an asymmetric cryptographic algorithm to obtain the security in message transmission. The public key can also be known by an attacker. Asymmetric cryptography can provide confidentiality, integrity and authentication information such as symmetric cryptography, but also provides a non-repudiation. Unfortunately, asymmetric cryptography is demanding the computing power. Applications using the asymmetric cryptography must be written for the devices with more computing power [15]. Another disadvantage for this model (asymmetric key encryption) is key management. In practice most attacks on public key systems will be aimed at the key management, rather than at the cryptographic algorithm(s) [16]. Private key must be store in owner's key-side such as mobile phone or bank server. Although key management in bank server-side may be secure but there is no guarantee for saving private key on mobile phones. For example the handset could be easily stolen or it might be hacked via Bluetooth.
Private key in mobile device can be stored either using file stored in JAR or record stored in RMS (Record Management System) [13]. File stored in JAR refers to storage of the private key in same JAR package as the application program along with the other class file. Record stored in RMS refers to use of a subsystem of the MIDP (Mobile Information Device Profile) in the J2ME (Java 2 Platform, Micro Edition) standard [14]. Unfortunately there are many free tools to decompile JAR file and edit class files, also there are many HEX editor tools for retrieving key from jar files. Besides, byte array data can be extracted from RMS easily. A further security enhancement provided by MIDP 2.0 enabled phones is the verifying of the cryptographically signed jar file on the mobile device, which gives users some security about executing downloaded code. In this way, the certificate encapsulated within the jar file is also verified so that man-in-the-middle attacks can be prevented [1].

## III. The Novel SMS Banking Model

### *A. Basic concepts of the proposed model*

This model is based on the following assumptions:

- The user is authenticated by bank server for once. This authentication means that service provider (bank server) store PIN (Personal Identification Number), IMEI (International Mobile Equipment Identity) and IMSI (International Mobile Subscriber Identity) of users. In first connection or better to say in registration process the user is required to self-select a PIN. This is the user

selected password that only the user and bank server should know. This information can be transferred to bank server in first connection by the customer at the counter or automatically (via SMS). It is clear that the first method is more secure, because in first connection there is no secure channel and encryption of the message.

- The users trust the bank server.
- The proposed model provides a secure communication between the user (mobile phone) and the bank server. Therefore, it is assumed that the bank server has secure communication channels between application server and backend database.
- As previously explained, PIN is a password that only the user and bank server know and PIN must not be stored on user's mobile phone.
- It is assumed that the bank has a secure method to distribute the mobile banking application to the user's mobile phone.
- It is assumed that the user's mobile phone is compatible with the mobile application. The mobile phone must be able to run the application or else the user cannot perform mobile banking via SMS.

In the proposed model the text message that contained customer account number information and other information such as the customer's account secret number will be entered into symmetric encryption algorithm and encrypted message as result of encryption module is achieved. The encryption key is obtained from key generation module in client-side (the user) and server-side. Moreover this operation, for inclusion integrity, the text message is entered to the hash function and hashed message is generated. The generated hashed message will be added to the end of encrypted message and finally encrypted message is ready to be delivered to GSM network. After transmitting the message through mobile network and receiving by the bank, the bank server checks IMSI of the received message with user's information in registration process. If IMSI was not in the list, server rejects the message. This strategy is the first step for prevention of the attacker to endangering the server bank with random messages. Then the sender's information such as PIN and IMEI corresponding to IMSI will be derived. With these information and time information, the server can produce the key and eventually decryption operations are done. Details of this process are described as follows:

*B. Key generation module*

In the proposed protocol, key generation is based on the use of an OTP (One Time Password) function. OTP is a password that oppose to traditional password only active during a distinct session or transaction. In other words the password generator function generate different password based on different time. The point is that these algorithms based on input parameters, generate OTP randomly as could not be guessed. The OTP generation algorithms typically generate OTP upon three approaches below:

- Based on time synchronization between two parties (password generators): These passwords are valid only for a short period of time. In this method the generated passwords are constantly changing (like every few minutes). For the generation of an equal password, both sides should be in the same time. Note that the problems arise from sync issues will be discussed further.
- Using a mathematical algorithm to generate new password based on previous password: In this method, the complex mathematical algorithm like cryptographic hash function is used for password generation based on previous password. Hash functions are one-way functions, so finding the previous password is extremely difficult to do and it is a computationally infeasible task.
- OTP is according to a mathematical algorithm for generating password based on challenge-responses.

In mathematical approach, restriction factors such as processor, memory and battery, limit the using of password generation algorithm based on complex mathematical algorithm. Moreover in second approach we need to store previous password for generating a new password and this is a insecurity point, on the other hand in time-sync algorithm there is another challenge and that is synchronization of time between both sides. As we go further we will describe how the proposed model will overcome this problem easily.
Generally in bank applications, there are two methods for generating and using an OTP:

- Connection-less OTP: in this method both sides have OTP generator function's parameters and can generate password independently. Whereas time is one of the parameters, for achieving same key or password, both sides must be time synchronized or almost the same time.
- Requesting OTP via SMS: While for any reason such as non time-sync or limitation in resource in one side, calculation and generation of OTP is not feasible; one side (usually the user/client) send request to other side (usually the bank-server/server) for OTP. In this state no OTP is generated locally. For authentication in server side, client sends special and unique information by SMS. Then server checks the content of the received SMS and if authenticity of information was valid, it generates an OTP randomly and sends to client by SMS. The client has a finite time to use this OTP. Afterwards, the transmitted OTP has no validity and cannot be used. This method have two major disadvantages: the first one is because of the additional cost imposed as sending and receiving of at least two SMS (one from client and the other from server) is needed. The second disadvantage is due to the none encrypted plaintext of the requested OTP and this is a point of vulnerability.

The input parameters of OTP in the proposed model are as follows:

- **IMEI (International Mobile Equipment Identity):** is an unique code for identifying mobile phones. This code can be exploited from mobile phones and stored in bank's server for each customer.

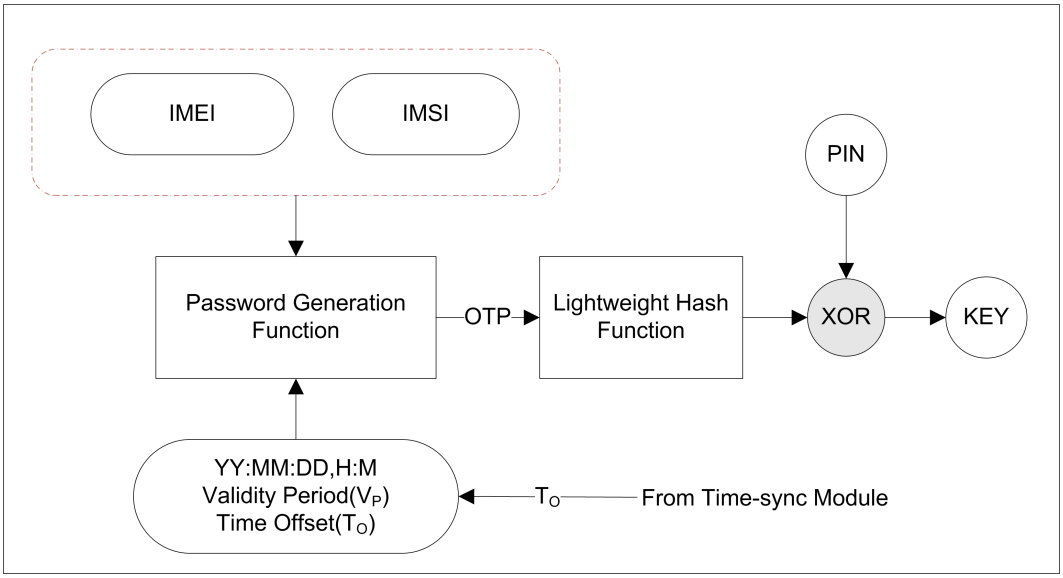


**Figure. 2**: The proposed Model for key generation module

- **IMSI (International Mobile Subscriber Identity):** is a unique number associated with all GSM network mobile phone users. In simple term, it is the mobile phone number. It is stored as a 64 bit field in the SIM, inside the phone and is sent by the phone to the network. This code is also stored in bank server per customer.
- **PIN (Personal Identification Number):** for authenticity of mobile phone holders we used another factor that is like a password and produce by the user ( customer). With this way even if mobile phone is stolen, theif can not generate OTP without the PIN. This password is similar to the regular passwords that user uses in the PCs, ATMs and must be carefully chosen to have enough complexity for preventing brute-force attacks and password guessing. It is necessary to mention that PIN will be stored in bank server in registration process.
- **YY:MM:DD,H:M :** represents date and time in OTP generator side.
- $V_P$ **(Validity Period):** This is OTP validity time interval. This time is a factor of network traffic for sending and receiving SMS and also time between receiving SMS in the bank server and processing messages. This time can be different from one up to few minutes. The processing time in server-side will increase when a high volume of requests sent to bank server, server will be forced to put requests in a queue and process them one by one.
- $T_O$ **(Time Offset):** It is the time difference between both sides. The assumption is both sides are time synchronized in the first connection, thus $T_O = 0$. If there was no time synchronization, the time difference will be calculated by a mechanism that will be explained further. Then for generating a new OTP, new $T_O$ will be applied to OTP generation algorithm.

In the key production stage there are some inputs that must be gathered first. In the *client-side*, the IMEI, IMSI are derived from costumer's mobile phone; time parameters ($YY$, $MM$, $DD$, $H$, $M$) are derived from telecommunication network and $T_O$ (which is assumed to be zero in first connection) and $V_P$ (which is assumed to be ten minutes) enter to the OTP function generator to produce OTP. To exert more secure control and also achieving a key with constant length, the obtained OTP from previous step will be hashed with a hash function. As explained before, for authenticity of mobile phone holder we used PIN. So that PIN will be XORed with OTP and finally encrypt/decrypt key is ready. Figure 2 show the proposed model of key generation module. With this way we can be sure if even the mobile phone is stolen, any one could not be generate key without PIN.

In the client side which is shown in figure 3, plaintext is inputted to an encryption module. The security of this module is based on a light weight symmetric algorithm. Stream ciphers as part of the symmetric key cryptography family, have always had the reputation of efficiency in Software and speed. They have attracted much attention since the beginning of the eSTREAM project in 2004. In April 15, 2008, the eSTREAM competition was finished and according to the final report [17], HC-128 [24], Rabbit [23], SOSEMANUK [22], and Salsa20/12 [21] were selected as the finalists of software profile of eSTREAM project. Salsa20/12 is the selected primitive for the encryption of the SMS banking model in the client side.

Salsa20 is a stream cipher introduced by Bernstein in 2005 as a candidate in the eSTREAM project. Bernstein also submitted to public evaluation the 8- and 12-round variants Salsa20/8 and Salsa20/12 [25], though they are not formal eSTREAM candidates. Later he introduced ChaCha [26], a variant of Salsa20 that aims at bringing faster diffusion without slowing down encryption. The compression function Rumba [27] was presented in 2007 in the context of a study of generalized birthday attacks applied to incremental hashing, as the component of a hypothetical iterated hashing scheme. Rumba maps a 1536-bit value to a 512-bit (intermediate) digest, and Bernstein only conjectures collision resistance for this function, letting a further convenient operating mode provide extra security properties as pseudo-

randomness. In order to achieve the message integrity a light weight hash function is needed. So Salsa20/12 is an all-in-one choice for our novel SMS banking model as we can use Rumba compression function for the Hashing part of the key generation module. Now for the clearity we give a concise description of the stream ciphers Salsa20 and the compression function Rumba.

*1) Salsa20*

The stream cipher Salsa20 operates on 32-bit words, takes as input a 256-bit key $k = (k_0, k_1, ..., k_7)$ and a 64-bit nonce $v = (v_0, v_1)$, and produces a sequence of 512-bit keystream blocks. The $i_{th}$ block is the output of the Salsa20 function, that takes as input the key, the nonce, and a 64-bit counter $t = (t_0, t_1)$ corresponding to the integer $i$. This function acts on the $4 \times 4$ matrix of 32-bit words written as

$$X = \begin{pmatrix} x_0 & x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 & x_7 \\ x_8 & x_9 & x_{10} & x_{11} \\ x_{12} & x_{13} & x_{14} & x_{15} \end{pmatrix} = \begin{pmatrix} c_0 & k_0 & k_1 & k_2 \\ k_3 & c_1 & v_0 & v_1 \\ t_0 & t_1 & c_2 & k_4 \\ k_5 & k_6 & k_7 & c_3 \end{pmatrix}$$

The $c_i$s are predefined constants. There is also a mode for a 128-bit key $k'$, where the 256 key bits in the matrix are filled with $k = k'||k'$. If not mentioned otherwise, we focus on the 256-bit version. A keystream block $Z$ is then defined as

$$Z \quad = \quad X \oplus X^{20}$$

where $+$ symbolizes wordwise integer addition, and where $X^r = Round^r(X)$ with the round function Round of Salsa20. The round function is based on the following nonlinear operation (also called the quarterround function), which transforms a vector $(x_0, x_1, x_2, x_3)$ to $(z_0, z_1, z_2, z_3)$ by sequentially computing

$$\begin{aligned} z_1 &= x_1 \oplus [(x_3 + x_0) <<< 7] \\ z_2 &= x_2 \oplus [(x_0 + z_1) <<< 9] \\ z_3 &= x_3 \oplus [(z_1 + z_2) <<< 13] \\ z_0 &= x_0 \oplus [(z_2 + z_3) <<< 18] \end{aligned}$$

In odd numbers of rounds (which are called columnrounds in the original specification of Salsa20), the nonlinear operation is applied to columns $(x_0, x_4, x_8, x_{12})$, $(x_5, x_9, x_{13}, x_1)$, $(x_{10}, x_{14}, x_2, x_6)$, $(x_{15}, x_3, x_7, x_{11})$. In even numbers of rounds (which are also called the rowrounds), the nonlinear operation is applied to the rows $(x_0, x_1, x_2, x_3)$, $(x_5, x_6, x_7, x_4)$, $(x_{10}, x_{11}, x_8, x_9)$, $(x_{15}, x_{12}, x_{13}, x_{14})$.

*2) Rumba*

Rumba is a compression function built on Salsa20, mapping a 1536-bit message to a 512-bit value. The input $M$ is parsed as four 384-bit chunks $M_0, ..., M_3$, and Rumba's output is

$$\begin{aligned} Rumba(M) &= F_0(M_0) \oplus F_1(M_1) \oplus F_2(M_2) \oplus F_3(M_3) \\ &= (X_0 + X_0^{20}) \oplus (X_1 + X_1^{20}) \\ &\oplus \quad (X_2 + X_2^{20}) \oplus (X_3 + X_3^{20}) \end{aligned}$$

where each $F_i$ is an instance of the function Salsa20 with distinct diagonal constants. The 384-bit input chunk $M_i$ along with the corresponding 128-bit diagonal constants are then used to fill up the corresponding input matrix $X_i$. A single word $j$ of $X_i$ is denoted $x_{i,j}$ . Note that the functions $F_i$ include the feedforward operation of Salsa20. The diagonal constants for Salsa20 and for Rumba (functions $F_0$ to $F_3$) is shown in Table 2.

*Table 2*: $c_i$ predefined constants

| | Round | $F_0$ | $F_1$ | $F_2$ | $F_3$ |
|---|---|---|---|---|---|
| $c_0$ | 61707865 | 73726966 | 6f636573 | 72696874 | 72756f66 |
| $c_1$ | 3320646E | 6d755274 | 7552646e | 6d755264 | 75526874 |
| $c_2$ | 79622D32 | 30326162 | 3261626d | 30326162 | 3261626d |
| $c_3$ | 6B206574 | 636f6c62 | 6f6c6230 | 636f6c62 | 6f6c6230 |

In *server-side* as shown in the figure 4 after receiving SMS from network, IMSI will be derived and corresponding customer record will be retrieved. This record was previously produced in the registration process. The retrieved information contains the IMEI and PIN. With these parameters and time parameters ($YY$, $MM$, $DD$, $H$, $M$) and the same OTP algorithm the decryption key could be derived. If decryption key was correct, the cipher message received by SMS will be decrypted and for integrity assurance will be hashed and compared with received hash at the end of cipher message. If two hashes were equal then decrypted message is accepted as validated and authenticated request and will be sent to bank backend database for responding to query such as bill, money transfer and so on.

In case, the generated key in the client-side and server side does not match, due to the lack time synchronization then, the decryption algorithm cannot be able to decrypt cipher message correctly and probably a fake digest message will be produced. For the diagnosis of this issue a signature in the message can be inserted. This signature can be a word in begin/end of the message. Then the bank server sends its own time parameters to the client/user but not in plaintext format. Because time is one of the parameters for OTP generator function and finally encryption/decryption key production and compromising is a vulnerability point in the system security and hackers will be one step closer to achieving the key.
To overcome this problem in the proposed model these time parameters will be XORed with ciphertext and results are sent to the user via SMS. Figure 5 shows the time synchronization module. It is assumed that the application software in both sides store plaintext message, cipher message and it's time parameters for $2V_P$ (which is equal to 20 minutes in our assumption). Therefore time of the bank-server side can be calculated by XORing the received message with cipher message. Then $T_O$ will be obtained from subtracting this time from time of the sent message. Afterwards all of the operations for generating OTP, key, encryption and performed hashing and encrypted message will be sent again. With this method and using secure channel, time synchronization is done and so in second try operation is successful.

This question might be asked that regarding the key gener-

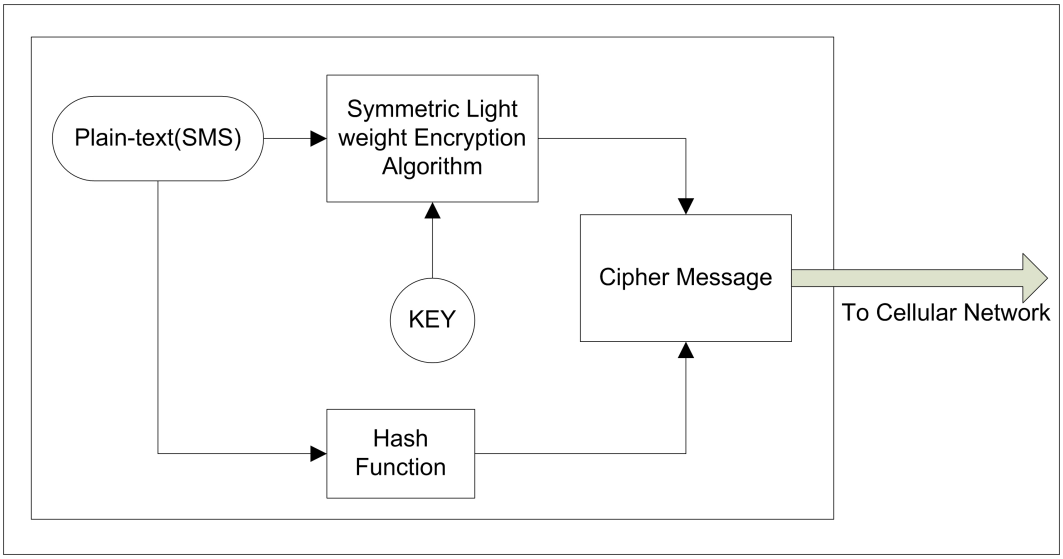


**Figure. 3**: The proposed Model for SMS sender

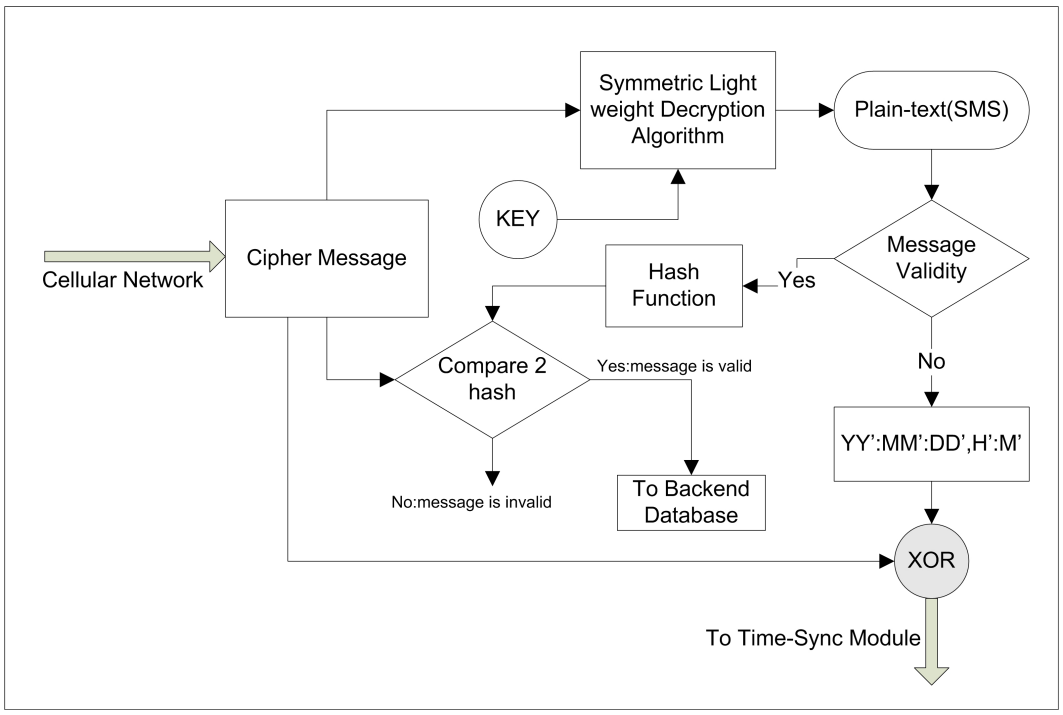


**Figure. 4**: The proposed Model in the Server-side (Bank server)

ation in both sides (the user/the bank server) which is based on obtaining time and date from telecommunication network, what is needed to calculate $T_O$? This is important since logically two parties must be time synchronized. In response to this question one can say when two parties are subscribers of common mobile operator the question might be correct but there is no guarantee for time synchronization between different mobile operators. On the other hand when the subscriber (the user) travel outside the geographical coverage area of the home network (the user is in roaming mode), time parameters will be obtained from local network. In this mode, issues such as GMT time differential and daylight saving will be added to impact of non time-synchronization. According to our proposed model can be assured that in time synchronization mode no additional SMS will not be exchanged but is worst case we can overcome the synchronization problem with exchanging maximum two SMS even in roaming mode. The full block diagram of the novel SMS Banking model is shown in fig. 6.

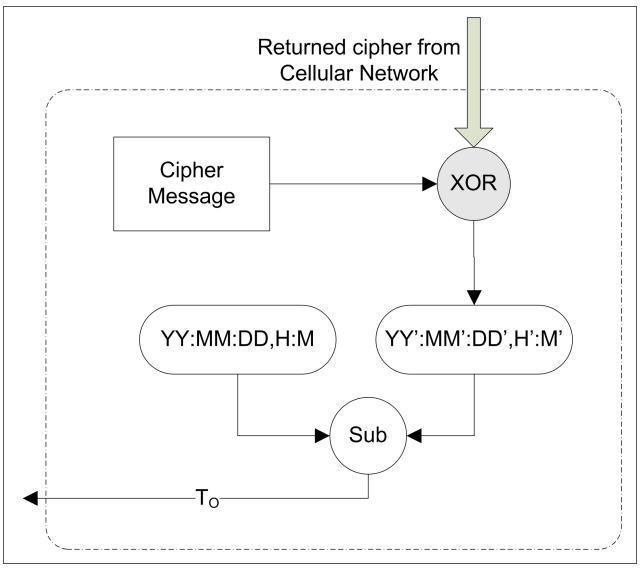


**Figure. 5**: Time syncronization module

## IV. Security Analysis of the Proposed Model

In this section, we perform security analysis of our proposed model. Current approaches of SMS banking systems use plaintext message for carrying out transactions. A highly motivated adversary can intercept this communication and gain access to modify important information. We observe that the goals of attacking an SMS banking system are to obtain the clients PIN and account number in order to fraudulently perform a banking transaction or simply read the balance or modify transfer information. In our system, the adversary can be successful if only he or she can intercept the message and get to know the encryption key. We perform security analysis of our proposed model according to principles of secure services. Protocol used in proposed model must be according to principles of secure services. These principles are: data confidentiality, data integrity, authentication, non-repudiation and availability [20].

### *A. Integrity*

The protocol employs a hashing algorithm to create a message digest of the message exchanged. The message digest is calculated both at the user-side and at the bank-server. This is performed by integrity check algorithm incorporated in SMS-banking application. If the content is altered during transmission a mismatch digest will occur and the receiver will know that the message has been compromised because there is very high probability that the output message digest will be different.

### *B. Confidentiality*

In our proposed model confidentiality is achieved by encrypting message using an OTP. As our assumption only the user and bank-server know the parameters for generation key. The level of security of the designed protocol depends on the strength of the encryption algorithm used.

### *C. Authentication*

For authentication purposes the protocol includes the clients account number and PIN. The user selects The PIN during registration for a bank account with the bank. The client enters his banking details that include the account and PIN in the mobile application and are used for authentication at the bank-server side. As PIN is not store on mobile phone the attacker cannot access the authentication detail (PIN) of the user therefore the attacker cannot use the authentication detail to perform masquerading attacks.

### *D. Availability*

The availability of our proposed model depends on three factors:

- Availability of the mobile phone: although Message decryption and calculating message digest can cost much of processing power but all selected algorithms for hash and encryption/decryption are light. Thus need to minimum resource for operation.
- Availability of the cellular network provider: If the cellular network is congested, the time to deliver the secured SMS message will be time-consuming.
- Availability of the bank server: Our proposed model guarantees minimum workloads of the server by discarding any message that causes the security verifications to return failed. This can decrease the workloads on the server side when the attacker tries to congest server with random messages. On the other hand availability of the banks service depends on the number of transactions that the server can handle at once. Number of transaction depends on the hardware capability. If the server hardware can handle multiple incoming messages then the server can perform multiprocessing to accommodate for more requests.

### *E. Non-repudiation*

Only the account holder and the bank server are supposed to have the required parameters for generating OTP and finally the key. Because OTP is obtained from IMEI, IMSI

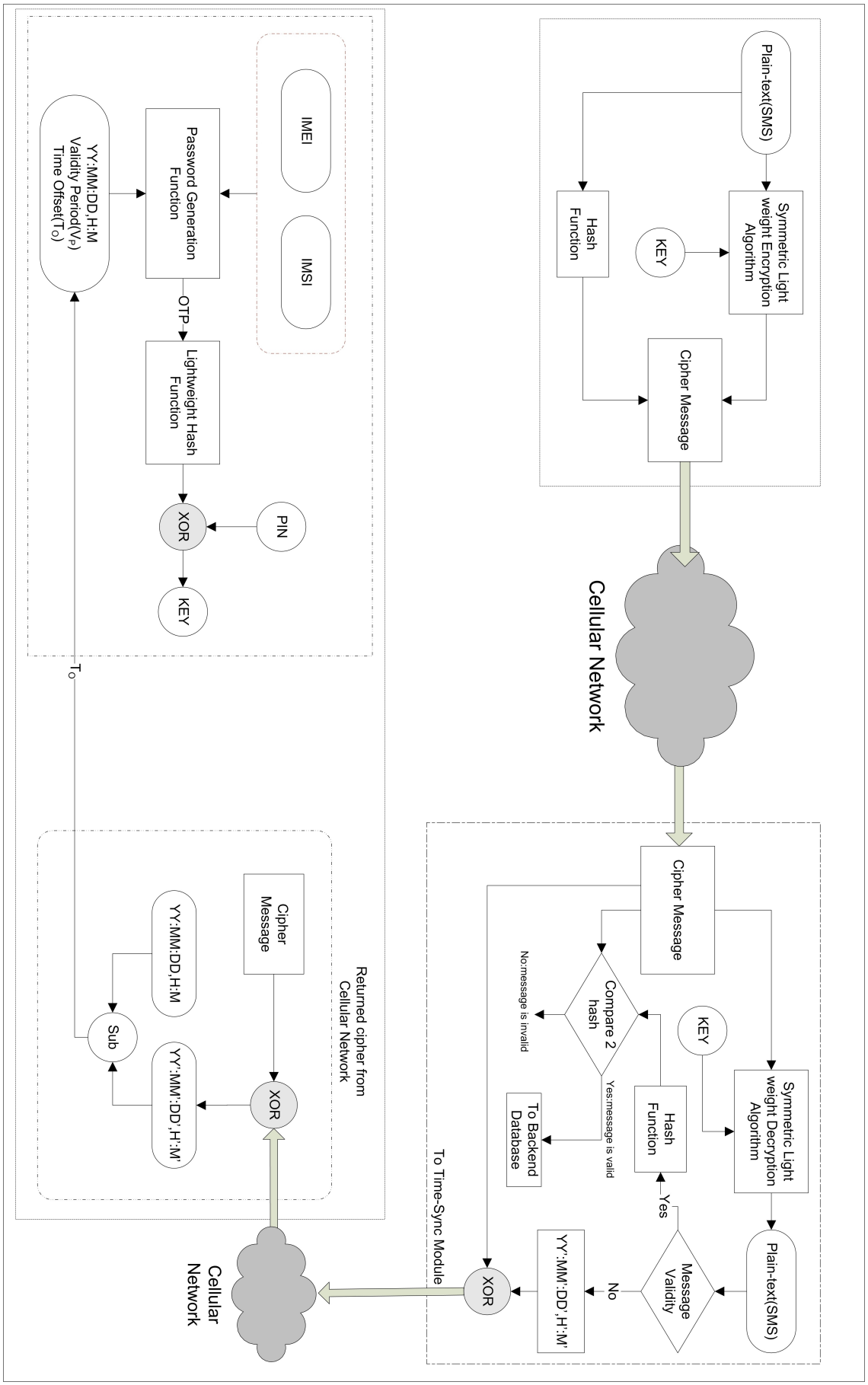


**Figure. 6**: The full block diagram of the novel SMS Banking model.

of the account holder's mobile phone and key is generated from OTP XOR PIN. Thus the user cannot deny not sending the message because only the specific user has the PIN and his/her mobile phone together. Even if the users claims the mobile phone is stolen, cannot deny sending the message as only he/she knows the PIN. Unless he/she claims forced to enter PIN.

## V. Conclusions

The main goal of this paper is to address the security shortfalls of SMS banking and propose a new SMS banking solution to fix those SMS banking vulnerabilities. The proposed model is in abstraction level and it is a new novel architecture for E2EE. While other proposed protocols and architectures do not pass resource constraint issues in mobile devices. Although symmetric key encryption is used in proposed model, but there is no need to exchange any keys. This is due to the novel OTP based key generation presented in the paper. With this idea there is no concern for the key management issues. On the other hand we have proposed a novel method for time synchronization through a secure channel that in all circumstances will be able to work.

## References


[1] J. Li-Chang Lo, J. Bishop, J.H.P. Eloff. SMSSec: An end-to-end protocol for secure SMS, *Computers & Security*, Vol. 27, 154 - 167, 2008.

[2] M. Rahnema. Overview of the GSM system and protocol architecture, *IEEE Communications Magazine*, Vol. 31(4), pp. 92 - 100, 1993.

[3] J. Golic. Cryptanalysis of Alleged A5 Stream Cipher, *Advances in Cryptography - EUROCRYPT 97*, LNCS 1233, pp. 239-255, 1997.

[4] A. Biryukov, A. Shamir, D. Wagner, Real Time Cryptanalysis of A5/1 on a PC, *Fast Software Encryption - FSE 2000*, LNCS Vol. 1978, pp. 37-44, 2000.

[5] T. Pornin, J. Stern. Software-Hardware Trade-offs application to A5/1 Cryptanalysis, *Cryptographic Hardware and Embedded Systems - CHES 2000*, LNCS Vol. 1965, pp. 155-184, 2000.

[6] M. Krause, BDD-based Cryptanalysis of Keystream generators, *Selected Areas in Cryptography - SAC 2007*, LNCS Vol. 4876, pp. 17-35, 2007.

[7] P., Ekdahl, T., Johansson. Another Attack on A5/1, *IEEE Transactions on Information Theory*, Vol. 49(1), pp. 284 - 289, 2003.

[8] T. Guneysu, T. Kasper, M. Novotny, C. Paar, A. Rupp. Cryptanalysis with COPACOBANA, IEEE Transactions on Computers, Volume 57, Issue:11, Page(s): 1498 - 1513, 2008.

[9] J. Hwu, S. Hsu, Y. Lin, and R. Chen, End-to-end security mechanisms for SMS, *International Journal of Security and Networks*, Vol. 1(3/4), pp. 177 - 183, 2006.

[10] K.F. Rafat, Enhanced text steganography in SMS, *2nd International Conference on Computer, Control and Communication, IC4 2009*, pp. 1 - 6, 2009.

[11] M. Shirali Shahreza, Stealth Steganography in SMS, *third IEEE and IFIP International Conference on wireless and Optical Communications Networks, (WOCN 2006)*, Bangalore, India, April 11-13, 2006.

[12] M.H. Shirali-Shahreza, M. Shirali-Shahreza, Steganography in SMS by Sudoku puzzle, *International Conference on Computer Systems and Applications, AICCSA'08*, pp. 844 - 847, 2008.

[13] J.E. Rice, Y. Zhu. A proposed architecture for secure two-party mobile payment, *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, PacRim'09*, pp. 88 - 93, 2009.

[14] E. Giguere, Databases and MIDP, Part 1: Understandingthe Record Management System, iAnywhere Solutions, February 2004.

[15] D. Lisonik, M. Drahansky. SMS Encryption for Mobile Communication, *International Conference on Security Technology*, pp. 198 - 201, 2008.

[16] RSA Laboratories. RSA Laboratories Frequently Asked Questions About Today's Cryptography, Version 4.1. RSA Security Inc., 2000.

[17] S. Babbage. The eSTREAM Portfolio, April 2008, eSTREAM project website.

[18] M. Boesgaard, M. Vesterager, T. Pedersen, J. Christiansen, O. Scavenius. Rabbit: A new high-performance stream cipher, Fast Software Encryption, Vol. 2887, Lecture Notes in Computer Science, pages 307-329. Springer, 2003.

[19] J. Aumasson, L. Henzen, W. Meier, M. Naya-Plasencia. Quark: A Lightweight Hash, 12th International Workshop Cryptographic Hardware and Embedded Systems, CHES 2010, LNCS, Vol. 6225, pages: 1-15, 2010.

[20] W. Stallings. *Network Security Essentials Applications and Standards*, Int. second edition, Prentice Hall, 2003.

[21] D. J. Bernstein. Salsa20/8 and Salsa20/12, eSTREAM, 2005, eSTREAM project website.

[22] C. Berbain, O. Billet, A. Canteaut, N. Courtios, H. Gilbert, L. Goubin, A. Gouget, L. Granboulan, C. Lauradoux, M. Minier, T. Ptonin and H. Sibert, SOSEMANUK, a fast software-oriented stream cipher, eSTREAM, 2005, eSTREAM project website.

[23] M. Boesgaard, M. Vesterager, T. Christensen, and E. Zenner, The stream cipher Rabbit, eSTREAM, 2005, eSTREAM project website.

[24] H. Wu, The Stream Cipher HC-128, eSTREAM, 2005, eSTREAM project website.

[25] D. J. Bernstein. Salsa20/8 and Salsa20/12. Technical Report 2006/007, eSTREAM, ECRYPT Stream Cipher Project, 2005.

[26] D. J. Bernstein. ChaCha, a variant of Salsa20. See http://cr.yp.to/chacha.html.

[27] D. J. Bernstein. What output size resists collisions in a XOR of independent expansions? ECRYPT Workshop on Hash Functions, 2007. See also http://cr.yp.to/rumba20.html.

[28] D. Saha, A. Mukherjee. Pervasive Computing: A Paradigm for the 21st Century, IEEE Computer Society Press, Vol. 36(3), pp. 25–31, 2003.

[29] D. Wagner, T. Pintaric, F. Ledermann, S. Dieter. Towards Massively Multi-User Augmented Reality on Handheld Devices, The Third International Conference on Pervasive Computing (Pervasive 2005), May 9-10, Munich, Germany, 2005.

fig1asasd

[30] R. Soram. Mobile SMS Banking Security Using Elliptic Curve Cryptosystem. International Journal of Computer Science and Network Security, Vol.9, No.6, pp. 30–38, 2009.

## Author Biographies

**Farzad Tavakkoli** is a Master of Science student at the University of Guilan in Iran. He received his B.Sc degree in Electronic engineering from the University of Guilan, Iran, in May 1997. He will receive his Master of Science degree from the University of Guilan, Rasht, Iran, in June 2011. He is worked as technical director in Information technology(IT) depatement many years.His research interests are in the areas of Mobile Computing, Information Security, distant Learning, Computer Networks, and M-Commerce. He may be reached at f.tavakkoli@msc.guilan.ac.ir.

**Reza Ebrahimi Atani** was born in 1980. He received the B.S. degree in electrical engineering from the University of Guilan in 2002 and the M.Sc and PhD degrees in electronics from Iran University of Science and Technology in 2004 and 2010 respectively. Since 2010, he is an assistant professor in Computer Engineering Department at the University of Guilan. His current research interests include stream cipher design and cryptanalysis, cryptographic hardware and embedded system (CHES), side channel attacks (Power and fault attacks), and design of VLSI circuits. He may be reached at rebrahimi@guilan.ac.ir (Corresponding Author).

**Shahriar Mohammadi** is a former senior lecturer at the University of Derby, UK. He also used to be a Network consultant in the UK for more than fifteen years. He is currently a lecturer in the University Of Khajeh, Nasir, Iran. His main research interests and lectures are in the fields of Networking, Data Security, Network Security, and e-commerce. He may be reached at smohammadi40@yahoo.com.