International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 5, No.1 (Jan-2016)

http://dx.doi.org/10.12785/ijcds/050106

Secure Hybrid Payment System in KSA

Aly M. Elsemary[1,2], Abdullah A. AlAnoor[1], Ahmad A. Alshammari[1], and Abdulaziz E. Aljohani[1]

[1]*Collage of Computer Science and Engineering, Taibah University, KSA.*
[2]*Systems and Computer Engineering Dept., Faculty of Engineering, Al-Azhar University, Cairo, Egypt.*

Received 30 Jun. 2015, Revised 5 Aug. 2015, Accepted 7 Nov. 2015, Published 1 Jan. 2016

Abstract: This paper introduces a new secure hybrid payment system in KSA that enables people to pay their bills, recharge their mobiles, as well as transfer balance using secure SMS messages. The proposed system implements a flexible and scalable architecture that has three main components: Customer, bank, and provider. The system is called hybrid because it combines two transmission media, SMS messages and the Internet. The SMS messages are used between the customer and the bank components while the Internet is used between the bank and provider components. Also the SMS messages can be used between the provider and the customer components. When a customer gets notified about his/her bill, he/she makes a payment request to the bank through a secure SMS message. The system supports both prepaid and post-paid payments. In prepaid, customers initiate payment requests with the desired amount while in post-paid, customers respond for payment notifications. Unlike current SMS-based mobile payment systems in which mobile operators are involved in transactions as a payment gateway, our system provides end-to-end secure transaction. Accordingly, the proposed system provides algorithms for key generation and distribution, confidentiality, integrity, authentication, non-repudiation, and protection against replay attacks. To complete the system, an android application called PayingSMS is developed to enable customers make their payment requests in an easy and secure environment. Finally, the security analysis of the proposed system is discussed.

Keywords: Payment system, Android application, Integrity, Confidentiality, SMS messages

1. INTRODUCTION

Payment systems as defined in [1] are implemented to enable the transfer of funds while defined in [2] as the set of processes and technologies that transfer monetary value from one entity or person to another. The improvements on these payment systems are provided by the Federal Reserve Banks [3]. The services provided by these payment systems are offered to several types of users including individuals, corporations, and governments.

The rapid growth and portability features of mobile phones make them practical to be combined in cashless payment mechanisms. Carat [4] presents mobile payment solutions to buy goods or services either through the prepaid phone card for low-value purchases or the monthly phone bill for both low and large amounts. These solutions have been presented as alternatives to several cashless payment systems [5, 6]. Contactless payments occur without physical connectivity between the reader and the cardholder device. A variety of technologies can make this happen including Bluetooth, infrared, and radio frequency identification (RFID). These technologies have been adapted by numerous quick-service oriented companies such as public transport (e.g., Octopus [5]) and fast-food restaurants (e.g., McDonalds). They are becoming the newest trends in the payment world. Currently, most of the contactless payment systems are based on the RFID technologies [7] since they have several advantages. These advantages include their development for traditional cash-based markets intended for small-ticket items such as items with $50 or less. In addition the RFID-based transponders are not powered by a battery. In contrast, the RFID payments have the shortcomings that Merchants have to install either new readers or attachments that are capable of reading an RFID transponder.

Like RFID-based payments, mobile-based payments provide contactless payments through the Internet. However, the Internet is not available for all people. Thus, SMS-based mobile payment systems are developed to achieve payments through SMS messages. In most of the current SMS-based payment systems [8, 9], a mobile

E-mail: {aelsemary@taibahu.edu.sa, alyelsemary@{azhar.edu.eg, ieee.org}} abo-hani1990@taibahu.edu.sa aes2009@taibahu.edu.sa ahmadxx@taibahu.edu.sa

operator acts as a payment gateway. The gateway intermediates the transaction between a customer and a merchant or a service provider. This means that customers have to subscribe with the mobile operator for services including making payments. Accordingly, a subscribed customer will be authorized to make phone calls including purchase products or services within certain credit limits. In this case, the mobile operator is responsible for transferring the purchased amount to the merchant or the service provider through the GSM network. Therefore, the security of the transaction depends on the security of the GSM network. Even though the GSM network supports traffic encryption between a mobile device and a base station, the implemented cryptographic algorithms are very weak and vulnerable to attacks as reported in [9]. Consequently, we proposed a new secure hybrid payment system to be applied in KSA. The proposed system provides end-to-end security to overcome the shortcomings of the most current SMS-based mobile payment systems. It supports confidentiality, integrity, authentication, and non-repudiation. In addition, it protects against replay attacks which are not considered by most of the current payment systems. The rest of this paper is organized as follows. Section 2 introduces related work while Section 3 presents the methodology that will be followed throughout this paper. Section 4 introduces the proposed secure hybrid payment system in KSA. Section 5 discusses the security provided by the proposed system. Finally, Section 6 concludes the paper and provides the future work.

2. RELATED WORK

Scientists make a great effort to develop SMS-based mobile payment systems. Harb *et al* [8] introduced a SMS-based mobile payment system called *SecureSMSPay*. The system uses mobile operators as a payment gateway. The gateway is responsible for transferring the information of payment amount between the banks of payer and payee. The authors provided security from the payers/payees to their banks while the security of the payment transfer from payer's bank to payee's bank through the gateway depends on the GSM network. The security of the GSM network is questionable as reported in [9]. Specifically, the encryption algorithm used by the GSM networks generates a new session key based on performing a hash function to one-bit-left shift to the current value of symmetric key [10]. The protocol has several weaknesses including 1) several fields in messages are sent in clear text such as payer and payee's mobile numbers. 2) The system changes session keys based on applying a hash function on the value of cyclic 1-bit-left shift of the current. Generating session keys in this way does not significantly increase security. 3) The payee or merchant initiates the payment request so that the system is hard to make a prepaid payment.

Toorani *et al* [9] presented an SMS-based payment system that supports secrecy, integrity, authentication, and non-repudiation. The protocol has the capability of verifying each party's public key and forward secrecy. As the system is based on public-key cryptography, it requires a trusted third party to act as a certificate authority. In addition, the protection against message replay attacks is not considered.

Duangphasuk *et al* [11] proposed a new payment protocol called SOMP which stands for SMS-based operator-assisted mobile payment. SOMP enables clients to perform payment transactions directly to the mobile operator itself or to merchants via the mobile operator. The authors satisfied necessary security functions through performing accountability analysis. Unfortunately, the authors did not show how they provide protection against message replay attacks.

Hashemi *et al* [12] developed a framework for secure SMS-based mobile payment. They describe in more details the interaction between involved parties including SMS gateway and Short Message Service Center. In addition, the description of several types of payment schemes for SMS-based payment was introduced. The authors implemented symmetric cryptography to secure transaction between engaging parties. Unfortunately, the shared keys between a client and his/her bank are distributed only when the client registers for the service at the first time. In addition, they did not describe how the session key is updated or generated.

Soni [13] developed a mobile payment model called *M-payment* between banks Using SMS. The *M-payment* model provides a direct link between banks and mobile operators (*i.e*., the third-party gateway is removed). Even though Soni provided a simple payment model based on SMS messages, he did not show how the security properties will be applied.

Saxena and Chaudhari [14] introduced a secure SMS protocol for VAS (numerous value added services) and other applications. Their protocol is called *secureSMS* which provides confidentiality, integrity, authentication, and non-repudiation. In addition, it protects against several attacks such as replay attacks. Even though their proposed model satisfies security properties, it is not designed specifically to payment applications but it needs to be adapted for this purpose.

Accordingly, we proposed a secure hybrid payment system (SHPS) to be applied in the KSA. The proposed system overcomes most of the aforementioned shortcomings associated with the current SMS-based mobile payment systems. The proposed system not only satisfies the security properties but also enables clients or customers to make payment requests in a simple and easy

way. In addition, it enables customers to make their payment requests from anywhere at any time through secure SMS messages. The system implements the AES symmetric encryption algorithm [15] to provide confidentiality while uses both AES and secure hash function algorithm one (*SHA1)* [16] to support integrity. In addition, the algorithms in [17, 18] can be used to support authentication. Also we developed an algorithm for key generation and distribution. Furthermore, we provided an algorithm for message authentication code generation and verification.

3. METHODOLOGY

SMS messages can be lunched from anywhere at any time but they are not secure. The proposed secure hybrid payment system takes the advantages of SMS messages and overcomes their shortcoming to make secure payments from anywhere at any time. The system allows a customer to send a secure payment message to a server called SMS server placed at a bank or as a separate third party server. Once the SMS server receives a payment request from a customer, it validates the message and then processes the payment on behave of the customer. Specifically, the bank component will transfer the amount due from the customer bank account into the provider/merchant bank account. Finally the SMS server notifies both the provider and the customer about the transfer process.

4. PROPOSED SHPS

The proposed new secure hybrid payment system (SHPS) assists people to make their payments from anywhere at any time in a secure environment. The SHPS deploys SMS messages in a secure environment to overcome the shortcomings associated with the current SMS-based mobile payment systems. We intend to apply the proposed system in the KSA (Kingdom of Saudi Arabia). The rest of this section introduces SHPS architecture, message formats conveyed among system components, database design, services provided by SHPS, and security implemented by SHPS.

A. SHPS Architecture

The system architecture of *SHPS* has three main components: *bank*, *provider* or *beneficiary*, and *user*. The *user* and *provider* components must register with the *bank* component. They have no direct communication between each other in the proposed architecture but the provider can notify customers about payments due regarding post-paid services. In contrast, the *bank* and *user* components interact via secure SMS messages while the *bank* and *provider* components interact through the Internet as depicted in Figure 1. Combining these two transmission technologies into the proposed system makes us refer to it as a hybrid system.

The *bank* component has two servers: database and SMS servers which are directly connected to each other through a network hub or switch. Note that the SMS server can be used as a separate third party server and accordingly, connected to the database server through the Internet. The database server contains all information related to customers and providers bank accounts. The SMS server is the main server of the SHPS. It is responsible for receiving and sending secure messages from/to the other components. It conveys the messages to the customer component via secure SMS messages while it exchanges messages with the database server and the provider component by sending secure network packets through the Internet. The SMS server has a database to hold all information related to registered users and providers welling to use the proposed payments. In addition, the database stores information about all performed transactions between *user* and *provider* components together with the database server.

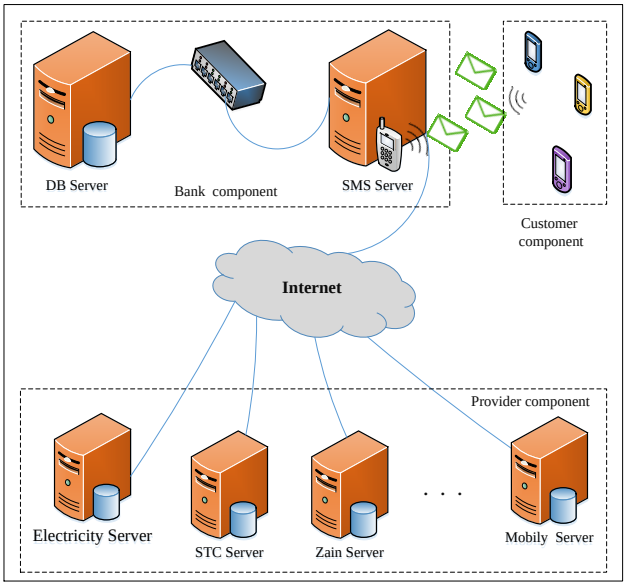


Figure 1: The proposed SHPS architecture.

The SMS server actually receives a payment request from a registered user. Next, it verifies the request against the security material assigned to the user. If the request is not valid in terms of security, it will be ignored. Otherwise, the SMS server creates a new balance transfer message with the amount received from the customer together with other related information such as the involved customer and provider IDs and their bank accounts. After constructing the message, the SMS server applies the security properties on the message and then sends it to the database server to transfer the specified amount from customer bank account to the intended provider bank account. Once the SMS server receives a replay or balance transfer notification from the database server, it notified both the customer and provider about

the performed actions. During the whole performed transaction, the SMS server stores its related information together with the applied security material as evidence in case of repudiation from any involved party.

The *provider* or *beneficiary* component consists of a set of servers belonging to registered providers, at least one server for each provided service. The server is used to store and manipulate accounts of customers that need the provided service. Also each provider must have at least a bank account with the bank. Examples of provided services include mobile services, energy services, and airlines booking services. In the KSA, the mobile services are provided by companies such as STC (Saudi Telecommunication Company), Zain, or Mobily while the energy service is provided by Saudi Electricity Company. When a provider server receives a payment notification from the SMS server, it verifies the notification. If the notification is not valid, it is ignored. Otherwise, the server updates the service account of the associated customer based on the service type which is either prepaid or post-paid. If the service is prepaid, the server updates the provided service according to the received payment amount. Otherwise, the server updates the service account of the customer based on the received payment amount. After updating, the server sends a secure notification to the SMS server.

The last component is the *customer* component which includes a set of smart mobiles through which customers can make their payment requests or receive related notification messages. The SHPS requires that each customer install the developed mobile application called *PayingSMS* on his/her mobile. This application allows customers to make their payment requests in an easy and secure environment. The payment requests can be performed for paying bills, recharging mobiles number, or transferring money to other accounts. Like a provider, each customer must have a bank account to be able to use SHPS.

B. SHPS Conveyed Massage Formats

This section is concerned with the formats of conveyed messages among the SHPS components as shown in Figure 2. These messages can be classified into three main categories: messages between *user* and *bank* components, messages between *bank* servers, and messages between *Bank* and *provider* components as shown in Figures 2a, 2b, and 2c, respectively. The messages conveyed from *bank* component to *customer* component are regular text messages and they do not need any special formats. Therefore, they are ignored here.

28 bits	4 bits	4 bits	80 bits	80 bits	1 bit	64 bits	64 bits	128 bits	128 bits
CID	MF	Provider	ID	MobNo	MN	Nonce	TS	MAC	EK_S

1) Adding SIM card number format

28 bits	4 bits	80 bits	64 bits	64 bits	128 bits	128 bits
CID	MF	MobNo	Nonce	TS	MAC	EK_S

2) Deleting SIM card number format

28 bits	4 bits	80 bits	16 bits	64 bits	64 bits	128 bits	128 bits
CID	MF	MobNo.	AM	Nonce	TS	MAC	EK_S

3) Recharging mobile number format

28 bits	4 bits	4 bits	72 bits	24 bits	64 bits	64 bits	128 bits	128 bits
CID	MF	Company	BillNo.	AM	Nonce	TS	MAC	EK_S

4) Paying bill format

28 bits	4 bits	128 bits	128 bits	24 bits	64 bits	64 bits	128 bits	128 bits
CID	MF	AN	BAN	AM	Nonce	TS	MAC	EK_S

5) Money transfer format

28 bits	4 bits	128 bits	64 bits	64 bits	128 bits	128 bits
CID	MF	AN	Nonce	TS	MAC	EK_S

6) Checking balance format

32 bits	128 bits	64 bits	64 bits	128 bits
CID	Password	Nonce	TS	MAC

7) Generating/Changing key format

a) Message formats from *user* to *bank* components.

4 bits	4 bits	128 bits	128 bits	24 bits	64 bits	64 bits	128 bits	128 bits
SMSSID	MT	AN	BAN	AM	Nonce	TS	MAC	EK_S

1) Money transfer format initiated by SMS server.

4 bits	4 bits	4 bits	64 bits	64 bits	128 bits	128 bits
DBSID	MT	Result	Nonce	TS	MAC	EK_S

2) Replay of money transfer format initiated by the DB server.

4 bits	4 bits	128 bits	32 bits	64 bits	64 bits	128 bits	128 bits
SMSSID	MT	AN	PIN	Nonce	TS	MAC	EK_S

3) Check balance format initiated by SMS server

4 bits	4 bits	128 bits	24 bits	64 bits	64 bits	128 bits	128 bits
DBSID	MT	AN	AM	Nonce	TS	MAC	EK_S

4) Replay of check balance format initiated by the database server.

b) Message formats between bank servers.

4 bits	4 bits	80 bits	80 bits	64 bits	64 bits	128 bits	128 bits
SMSID	MC	ID	MobNo	Nonce	TS	MAC	EK_S

1) Check a SIM card number format initiated by SMS server.

4 bits	4 bits	4 bit	64 bits	64 bits	128 bits	128 bits
PID/CID	MC	status	Nonce	TS	MAC	KS

2) Replay from provider format for checking.

4 bits	4 bits	80 bits	80 bits	24 bits	64 bits	64 bits	128 bits	128 bits
SMSID	MC	ID	MNorBN	AM	Nonce	TS	MAC	KS

3) Recharge/Bill-pay notification format initiated by SMS server.

4 bits	4 bits	4 bits	64 bits	64 bits	128 bits	128 bits
PID/CID	MC	Reason	Nonce	TS	MAC	KS

4) Recharge/Bill-pay confirmation initiated by provider/company.

c) Message formats between *bank* and *provider* components.

Figure 2. Conveyed message formats in SHPS.

On the other hand, the communication from customer to bank components requires secure formatted messages shown in Figure 2a. this group of messages has seven different formats: 1) adding SIM card number, 2) deleting SIM card number, 3) recharging mobile number, 4) paying bill, 5) transferring money, 6) checking balance, and 7) Generating/changing key. The first six formats share the same six fields: *CID*, *MT*, *Nonce*, *TS*, *MAC*, and EK_s. The *CID* refers to the customer identification while the *MF* indicates the message format. The codes identifying different message format *MF* are listed in Table 1. For example, the code 0011 refers to pay bill format. The *Nonce* is a unique random number for each message and time stamp *TS* refers to the current time. Both the *Nonce* and *TS* are used to protect against replay messages. The message authentication code *MAC* is used to verify the integrity of the message. Finally, the *EK*s is used to hold the encrypted session key k_s used to conceal the message.

The field *MN* in the first format of Figure 2a refers to whether the associated SIM card number is considered as a main number or not. Also the amount field *AM* indicates the recharging, billing, or transferring amount based on the used format. The *provider* and *ID* fields are used to hold the *provider* code and the identification number associated with the SIM card number in the *MobNo* field, respectively. The constructed codes for expected providers or companies are listed in Table 2. In the same way, the *BillNo* and *company* fields are used to hold the bill number and its beneficiary, respectively. The beneficiary/company codes are described in Table 2.

Table 1: Identify codes for different message formats.

MF	Description
0000	Refers to add SIM Card number format
0001	Refers to delete SIM Card number format
0010	Refers to recharge SIM Card number format
0011	Refers to pay bill format
0100	Refers to transfer money format
0101	Refers to check balance format
0110 to 1111	Reserved for future use

Table 2: Providers/companies assigned codes.

Provider/ company	Description
0000	Refers to Zain Company
0001	Refers to Mobily Company
0010	Refers to STC Company
0011	Refers to Saudi Electricity Company
0100	Refers to Saudi Airlines Company
0101	Refers to Altaiar Traveling Company
0110 to 1111	Reserved for future use

Also the fields *AN* and *BAN* are used to hold the account numbers from which and to which the amount *AM* is transferred, respectively. In addition, the filed *AN* refers to the account number for which the balance is checked. Finally, the password field is used to hold a random password when making a request to generate/change the encryption key *k* that is used in the encryption of the session key k_s.

In Figure 2b, the field *MT* indicates the message type conveyed between the SMS server and the bank database server. The different values of this field are described in Table 3.

Table 3: Codes of different *MT* between bank servers.

MT	Description
0000	Money transfer
0001	Replay of Money transfer
0010	Check balance
0011	Replay of check balance
0100 to 1111	Reserved for future use

The fields *SMSSID* and *DBSID* refer to the SMS and database servers, respectively. Each of these fields has four bits long which can accommodate up to 16 different SMS sever or database serves within the same bank. Finally, the *Result* field indicates the status of the money transfer transaction received from the database server. Table 4 describes the different *Result* codes that may take place.

Table 4: Result codes for money transfer.

Result	Description
0000	Money transferred successfully
0001	Failed since no enough money
0010	Failed since DB is out-of-service
0011	Failed since account restriction
0100 to 1111	Reserved for future use

In Figure 2c, the field *MC* refers to different message codes associated with messages conveyed between the *bank* and the *provider* components. It can be assigned one of the codes described in Table 5 (*e.g.*, the code 0101 refers to a replay message for bill payment). The *status* field indicates the status of the checking. Since this field has four bits, the code 0000 is assigned for valid number and the code 0001 is assigned for invalid number while the remaining fourteen codes are reserved for future use. Finally, the *Reason* field confirms the operation state accomplished by a provider or company server and it has the codes provided in Table 6.

Table 5: *MC* Codes between *Bank* and *provider* components.

MC	Description
0000	Check SIM card with provider
0001	Provider replay for check SIM card
0010	SIM recharge
0011	bill payment format
0100	Replay for SIM recharge
0101	Replay for bill payment
0110 to 1111	Reserved for future use

Table 6: Confirmation codes indicating operation state.

Reason	Description
0000	Successfully SIM recharged
0001	The SIM is not registered
0010	SIM card is no longer valid
0011	SIM card is temporary not available
0100	The bill amount is successfully received
0101	The customer account is temporary not available
0110 to 1111	Reserved for future use

C. SMS Server Database

The main database of the proposed SHPS is stored at the SMS server. The database has three main entities: customer, provider, and operation. These entities and their relations are described by the ERD (Entity Relationship Diagram) shown in Figure 3.

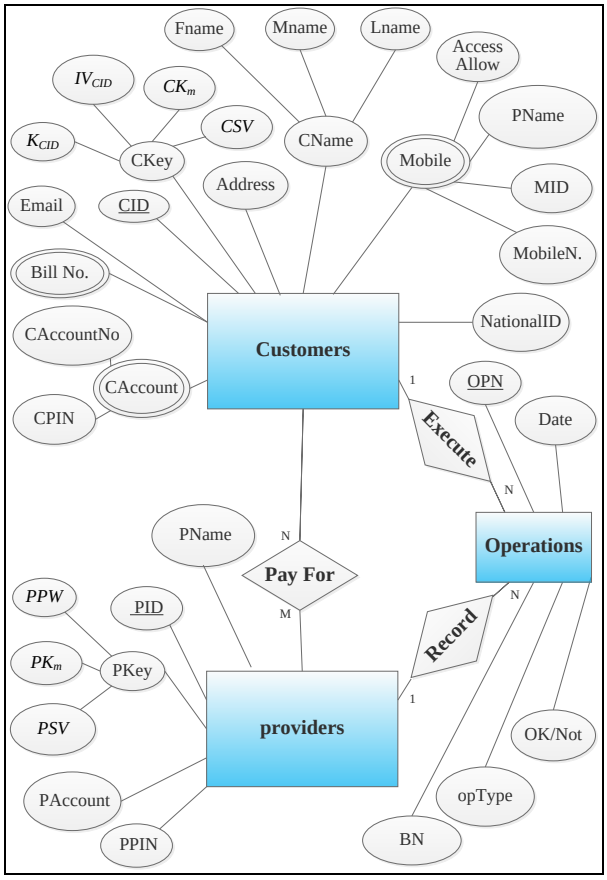


Figure 3. ERD of the SMS server database.

The customer entity has the attributes of registered customers. These attributes include name, address, bank account numbers, mobile numbers, master key (CK_m), secret value (*CSV*), customer password (*CPW*), and customer ID (*CID*). The CK_m, *CSV*, and *CPW* are secret materials used to secure messages communicated between the SMS server and the *customer*. The provider entity has attributes for providers or company. These attributes includes provider/company name, provider password (*PPW*), master key (PK_m), secret value (*PSV*) and provider account (*PAccount*). The *PPW*, PK_m, and PSV are secret materials used for securing conveyed messages between the SMS server and provider server. Finally, the operation entity stores information related to performed operations between customers and providers. The entity stores the time when an operation takes place, the type of the operation, the status of the operation, and the information of involved parties (i.e., customer and provider or company).

D. Services provided by SHPS

The SHPS provides several services which can be classified into three main categories: SIM card services, bill paying services, and bank transaction services. These service categories are discussed in more details in the rest of this section.

1) SIM Card Services

The SIM card services are concerned with all the services related to SIM cards. These services include adding a SIM card number, deleting a SIM number, and recharging a mobile number. Both add and delete SIM card services are used to add and delete a SIM card into or from the SMS server database, respectively. The SHPS requires that a SIM card be added before recharging it. Also the delete service is used to delete a SIM card when the number is no longer used. The recharge mobile number service allows a customer to recharge his/her mobile number with a specific amount. Due to the limited space, only the recharge SIM card service is discussed. A protocol of six steps is designed to handle the recharge process:

1. *CID* → *SMSSID*: *CID*, {*MF*, *MobNo*, *AM*, *Nonce*, *TS*, *MAC*}$_{ks}$, {k_s}$_{k1}$
2. *SMSSID* → *DBSID*: *SMSSID*, {*MT*, *AN*, *BAN*, *AM*, *Nonce*, *TS*, *MAC*}$_{ks}$, {k_s}$_{k2}$
3. *DBSID* → *SMSSID*: *DBSID*, { *MT*, *Result*, *Nonce*, *TS*, *MAC*}$_{ks}$, {k_s}$_{k2}$
4. *SMSID* → *PID*: *SMSID*, { *MC*, *ID*, *MNorBN*, *ID*, *AM*, *Nonce*, *TS*, *MAC*}$_{ks}$, {k_s}$_{k3}$
5. *PID* → *SMSID*: *PID*, { *MC*, *Reason*, *Nonce*, *TS*, *MAC*}$_{ks}$, {k_s}$_{k3}$
6. *SMSID* → *CID*: Notification SMS message with *Reason* translation.

The protocol uses six different message formats; the first five are from Figure 2 while the last one is a regular text message. All the variables in this protocol are discussed in the message formats in Section 3.2. The protocol says that the customer who has the *CID* send a recharge request message to the SMS server which is identified by *SMSSID*. The message contains all variables after the colon in each step. The variables between curly brackets are encrypted with the subscript key at the closed curly bracket. For example, the *CID* in step 1 generates a random session key (k_s) and uses it in the encryption of the string resulted from the concatenation of *MF*, *MobNo*, *AM*, *Nonce*, *TS*, and *MAC*. *Next*, k_s is encrypted with k_1 that is shared between the *CID* and the *SMSSID*.

To illustrate this protocol, suppose you want to recharge the pre-added mobile number *0563333333* which is associated with the *ID 1065477250*. The customer *ID* is 01234 while the provider corresponding to the mobile number is Mobily (i.e., the provider code is 0001, in this case). Also assume that the customer has the bank account number *abcd* while the provider has the bank account number *wxyz*. In addition, the mobile number will be charged successfully with the amount of 20 SR. Accordingly, the protocol will conveys these messages in the specified order as shown in Figure 4.

1. *CID* → *SMSSID*: 01234, {0010, *0563333333*, *20*, *Nonce*, *TS*, *MAC*}$_{ks}$, {k_s}$_{k1}$
2. *SMSSID* → *DBSID*: *SMSSID*, {0000, *abcd*, *wxyz*, *20*, *Nonce*, *TS*, *MAC*}$_{ks}$, {k_s}$_{k2}$
3. *DBSID* → *SMSSID: DBSID*, {0001, 0000, *Nonce*, *TS*, *MAC*}$_{ks}$, {k_s}$_{k2}$
4. *SMSID* → *PID*: *SMSID*, {0010, *1065477250*, *0563333333*, *20*, *Nonce*, *TS*, *MAC*}$_{ks}$, {k_s}$_{k3}$
5. *PID* → *SMSID*: *PID*, {0010, 0000, *Nonce*, *TS*, *MAC*}$_{ks}$, {k_s}$_{k3}$
6. *SMSID* → 01234: the mobile No. ********33 has been recharged successfully.

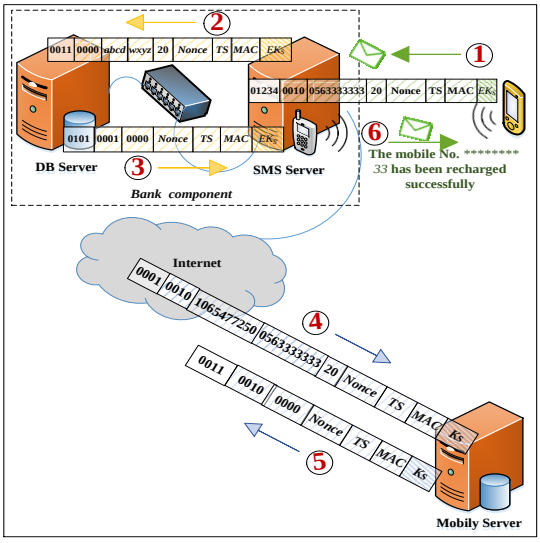
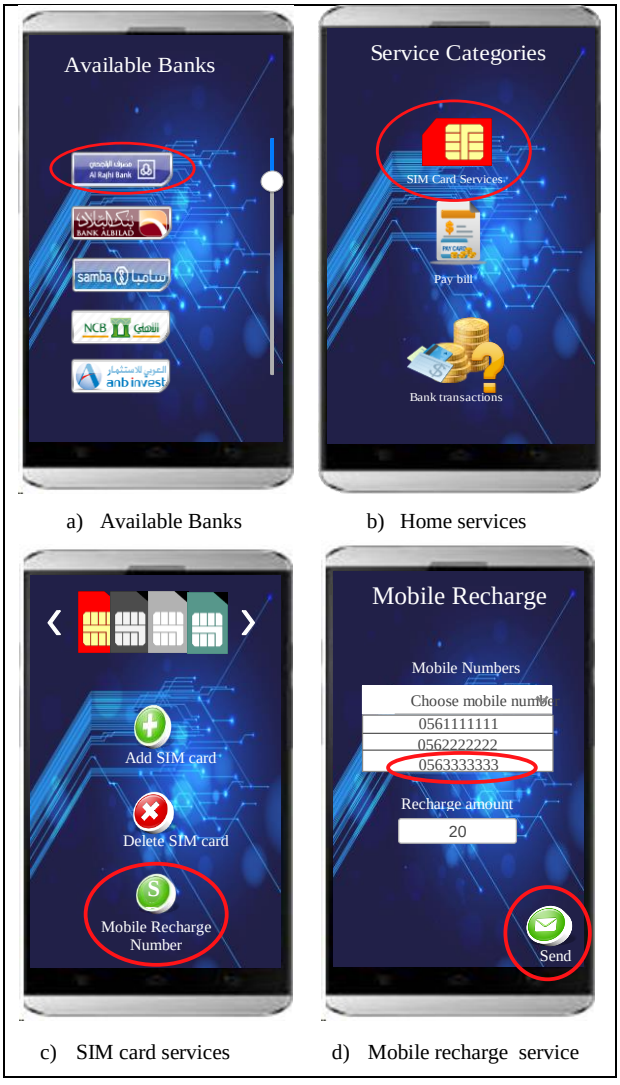


Figure 4. System interaction for successfully recharge the mobile.

The customer can lunch the recharge payment request through the developed *PayingSMS* application installed on his/her smart phone. The customer selects in order the bank from which the amount will be transferred, the *SIM card services*, *Recharge a SIM card*, and then the number 0563333333 as shown in Figures 5a, 5b, 5c, and 5d, respectively. Finally, the customer enters the recharge amount 20 and then press *Send* button. Note that when choosing a bank, the customer will be asked for its credential information. If it is correct, the *Home service* in Figure 5b will appear and the customer can continue to complete the transaction. Otherwise, the transaction cannot be continued.



a) Available Banks b) Home services

c) SIM card services d) Mobile recharge service

Figure 5. Steps of recharging SIM card.

2) Paying Bill Services

The paying bill service allows customers to pay their bills or payments easily through just sending a secure SMS to the SMS server with the desired amount. The SHPS achieves the payment through the same protocol used for mobile recharge except that the first and the last steps use different message formats. Accordingly, the protocol steps become as follow:

1. *CID → SMSSID*: *CID*, {*MF*, *Company*, *BillNo*, *AM*, *Nonce*, *TS*, *MAC*}$_{ks}$, {k_s}$_{k1}$
2. *SMSSID → DBSID*: *SMSSID*, {*MT*, *AN*, *BAN*, *AM*, *Nonce*, *TS*, *MAC*}$_{ks}$, {k_s}$_{k2}$
3. *DBSID → SMSSID: DBSID*, { *MT*, *Result*, *Nonce*, *TS*, *MAC*}$_{ks}$, { k_s}$_{k2}$
4. *SMSID → PID*: *SMSID*, { *MC*, *ID*, *MNorBN*, *AM*, *Nonce*, *TS*, *MAC*}$_{ks}$, { k_s}$_{k3}$
5. *PID → SMSID*: *PID*, { *MC*, *Reason*, *Nonce*, *TS*, *MAC*}$_{ks}$, { k_s}$_{k3}$
6. *SMSID → CID's mobile*: Notification SMS message of paying the bill

To illustrate this protocol, Figure 6 is constructed with the assumption that the customer has the *CID* 01234 and bank account number *abcd* as well as Electricity company account *123456789*. Also the electricity bill due with 200 SR should be paid to the *Electricity Company* bank account *lmno* from the customer bank account. Accordingly, the customer sends the request message with the associated information to the SMS server as shown in Step 1 in Figure 6.

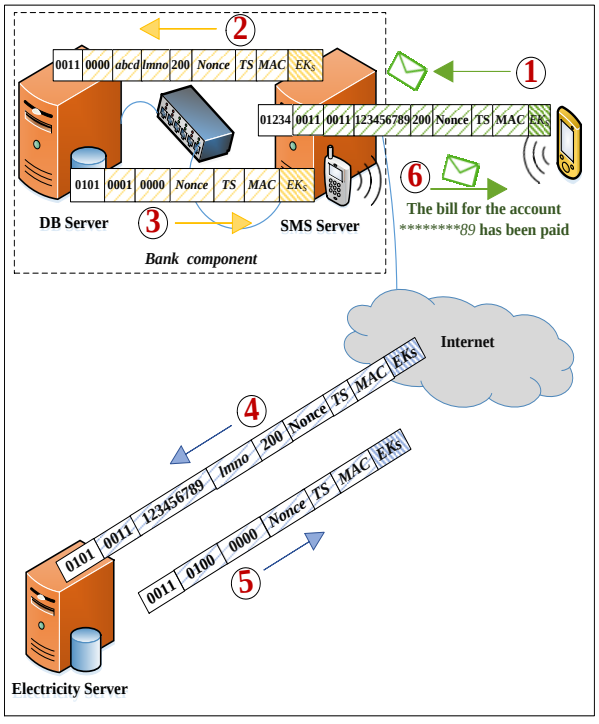


Figure 6. The visualization of the billing protocol.

Once the SMS server receives the request, it checks the message validity and then creates a related message to request the database server to transfer the specified amount 200 SR from the account *abcd* to the account *lmno*. Once the database server receives the message, it performs the request and returns a respond message to the SMS server which notifies both the provider and then the customer after receiving confirmation from the provider. According to the protocol steps, the following messages will be conveyed in order:

1. *CID → SMSSID*: *01234*, {*0011*, *0011*, *123456789*, *200*, *Nonce*, *TS*, *MAC*}$_{ks}$, {k_s}$_{k1}$
2. *SMSSID → DBSID*: *0011*, {*0000*, *abcd*, *lmno*, *200*, *Nonce*, *TS*, *MAC*}$_{ks}$, {k_s}$_{k2}$
3. *DBSID → SMSSID: 0101*, { *0001*, *0000*, *Nonce*, *TS*, *MAC*}$_{ks}$, { k_s}$_{k2}$
4. *SMSID → PID*: *0101*, { *0011*, *123456789*, *lmno*, *200*, *Nonce*, *TS*, *MAC*}$_{ks}$, { k_s}$_{k3}$
5. *PID → SMSID*: *0011*, { *0100*, *0000*, *Nonce*, *TS*, *MAC*}$_{ks}$, { k_s}$_{k3}$
6. *SMSID → CID's mobile*: The bill for the account ********89 has been paid.

3) Bank Transaction Services

The bank transaction services include balance check and money transfer services. The balance check service allows a customer to check the current amount in his/her bank account while the money transfer service enables a customer to transfer money from his/her bank account to a family or friend account. Since both of these services have similar behavior, only the money transfer protocol is presented. The protocol has four steps to enable a customer to transfer a specific amount from his/her account into another account. The four steps are described as the following:

1. *CID → SMSSID*: *CID*, {*MF*, *AN*, *BAN*, *AM*, *Nonce*, *TS*, *MAC*}$_{ks}$, {k_s}$_{k1}$
2. *SMSSID → DBSID*: *SMSSID*, {*MT*, *AN*, *BAN*, *AM*, *Nonce*, *TS*, *MAC*}$_{ks}$, {k_s}$_{k2}$
3. *DBSID → SMSSID: DBSID*, { *MT*, *Result*, *Nonce*, *TS*, *MAC*}$_{ks}$, { k_s}$_{k2}$
4. *SMSSID → CID's mobile*: Notification SMS message of balance transfer.

Steps 2 and 3 are the same steps in the paying bill and recharging mobile number protocols while steps 1 and 4 uses the money transfer request and notification message formats in Figure 2. To demonstrate the protocol, Figure 7 is constructed to visualize the message interaction during the money transfer process. The example assumes that the customer *CID* is 01234 and wants to transfer 1000 SR from his/her account *abcd* to a beneficiary account *pqrs*.

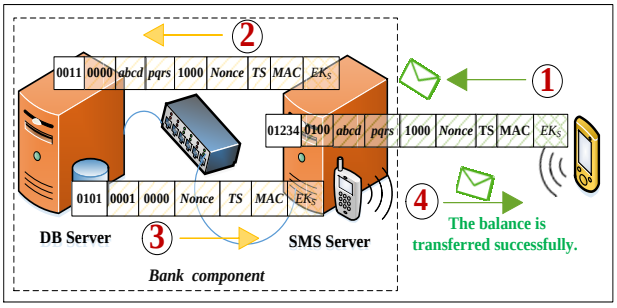
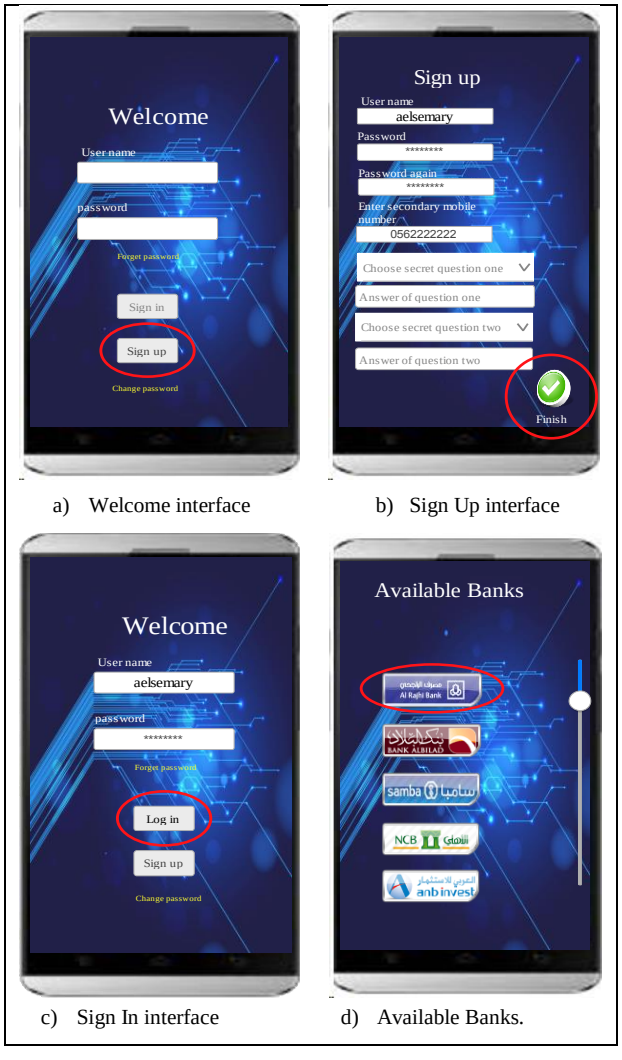


Figure. 7. Money transfer interaction.

E. *Security Implemented by SHPS*

The underlying SHPS satisfies the security properties through two layers of security. The first layer is concerned with access control for the developed *PayingSMS* mobile application installed on a customer mobile as shown in Figure 8.



a) Welcome interface b) Sign Up interface

c) Sign In interface d) Available Banks.

Figure 8. *PayingSMS* access control.

The application allows a customer to set up *a username* and a *password* on the entire application as depicted in Figures 8a and 8b. This layer only allows people who provide the correct credential information to access the application and navigate through the available bank interfaces as seen in Figures 8c and 8d. This layer is configured immediately after installing the *PayingSMS* application which enables customers to make secure requests to the SMS server.

The second layer of security is concerned with securing messages conveyed among the SHPS components. This layer satisfies the security goals via providing confidentiality, integrity, authentication, and non-repudiation. In addition, this layer protects against replay attacks. To achieve these security requirements, encryption keys must be generated and distributed. Also, a message authentication code (*MAC*) should be created. Thus, the rest of this section is dedicated to show how encryption keys are generated and distributed. In addition, the security environment in SHPS is introduced.

1) *SHPS Key Generation and Distribution*

The key generation and distribution process shows how an encryption key is generated and distributed between two of underlying system components. Even though this can be applied between any two components of the system, the rest of this section is devoted to generate and distribute a key between the *customer* and *bank* components. A customer should personally registers with the SMS server to obtain shared secret materials. These materials include a master key CK_m and a secret value *CSV*. The system assumes that these security materials are only known by both the customer and the SMS server. After obtaining these materials, the customer can generate or change an encryption key together with an initial vector and then distribute it to the SMS server. This is depicted in Figure 9 in which Figure 9a shows the key generation together with the initial vector and its distribution at the customer side while Figure 9b presents the same process at the SMS server.

To complete the process of a key generation and distribution between a customer and the SMS server, the customer should use the key generation service provided by the *PayingSMS* application. Accordingly, the service generates a password of sixteen letters as a combination of capital, small, number, and special letters. Next, this password is concatenated with the customer secret value *CSV* provided by the customer. Figure 9 refers to the concatenation by the symbol ||. The concatenation result is digested by the secure hash algorithm one SHA_1 which produces a unique hash value of 160-bit long for each message. The most significant 128 bits is used as the key K_{CID} while the least significant 32 bits is concatenated with the generated password. The concatenation result is

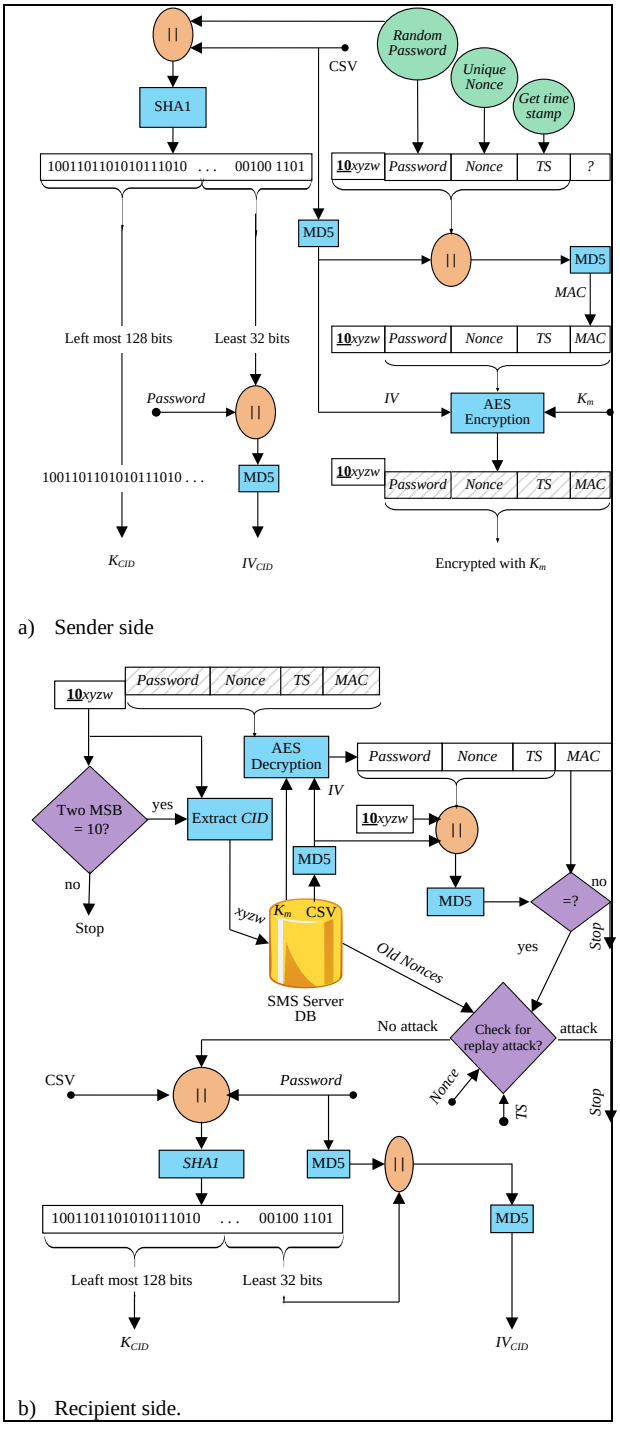


Figure 9. Key generation and distribution.

digested by MD5 algorithm [19] to produce the initial vector IV_{CID}. Both K_{CID} and IV_{CID} are associated with the customer identified by *CID*. The PayingSMS securely stores these values for later use during service requests. To distribute these values, the application sends the generated password to the SMS server which applies the same process to generate the same key and initial vector.

To send the password in a secure environment, the application, generates a unique nonce *Nonce* and obtains the current system time as a time stamp *TS*. Next, the *CID* is prefixed with **10** to notify the SMS server that the message is a key generation message. Then, the MAC value is created as the hash value of MD5 of the message that is obtained from concatenating in order the prefixed *CID*, *password*, *Nonce*, *TS*, and the digested value of CSV using MD5. Next, the message consists of the concatenation of *password*, *Nonce*, *TS*, and *MAC* is encrypted using the AES algorithm in CBC mode with the key CK_m and the initial vector *IV*. *IV* is generated as the digested value of CSV using MD5. Finally, the message composed of the concatenation of the prefixed *CID* (e.g., **10***xyzw*, in this case) and the resulted ciphertext is sent to the SMS server. The key is distributed by sending the whole message to the SMS server as depicted in Figure 9a in which *xyzw* refers to the value of the *CID* as an example. Note that **10** is prefixed to the value of CID (**10***xyzw*) to be recognized as a key generation message.

Once the SMS server receives the message, it recognizes the message as a key generation message because the *CID* is prefixed with the value **10**. Next, the *CID* is extracted by removing the attached prefix as shown in Figure 9b. Then, the SMS server indexes its database with the received *CID* (in this case, *xyzw*) to get the CK_m and *CSV* associated with the *xyzw*. Then, the message without the prefixed *CID* is decrypted with AES in CBC mode using CK_m and *IV* to recover the password. The *IV* is also obtained in the same way by digesting the customer secret value *CSV* using MD5. After recovering the message, the SMS server check the integrity by comparing the received *MAC* with MAC1 generated in the same way as *MAC* at the sender side. If the SMS server assured the integrity, it starts to check against replay attacks by matching the received *Nonce* with a set of old nonces within the time stamp *ST*. If the SMS server assured that the message is a refresh one, it will use the received *password* and the *CSV* indexed from the database to produce the K_{CID} together with IV_{CID} in the same way as in the sender side.

2) Security Environment is SHPS

The proposed system SHPS achieves the security of a message by applying integrity, authentication, and confidentiality together with protection against replay attacks. To facilitate the discussion, Figure 10 is constructed on the message format of paying bills described in the fourth format in Figure 2a. Figure 10a applies the security while Figure 10b verifies the applied security.

After the message is constructed, a nonce *Nonce* and a timestamp *ST* are obtained to protect against replay attacks. To protect the message integrity, a *MAC* is

generated as the encryption of a hash value using AES in the ECB mode with the key K_{CID}. The hash value is resulted from digesting a message using MD5. The message is constructed from concatenating all fields of the underlying message format except the *MAC* and EK_s as depicted in Figure 10a.

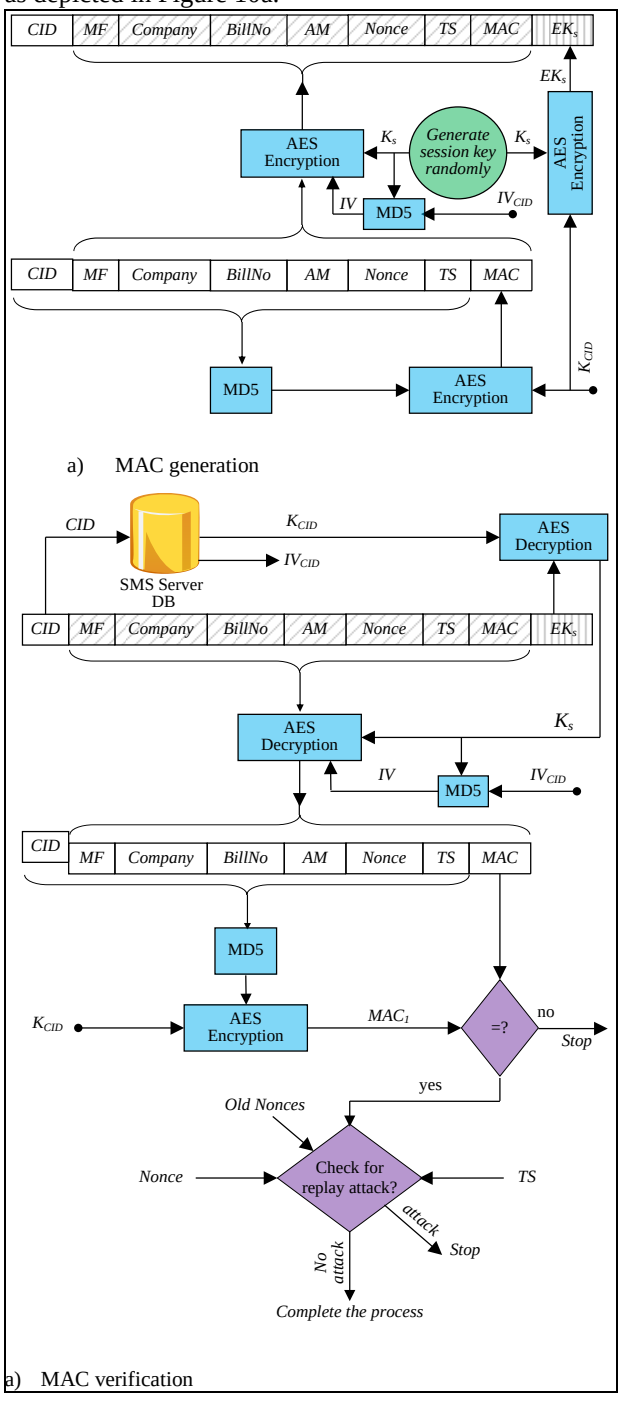


Figure 10. MAC generation and verification in SHPS.

To achieve confidentiality, the message is encrypted using AES in CBC mode using a session key K_s and *IV*. The session key, K_s, is uniquely generated for each message. The *IV* is obtained by digesting the concatenation of K_s and IV_{CID} using MD5. Since the *IV* depends on the K_s, it will also be unique for each message. Next, the session key (i.e., K_s), is encrypted using the AES in ECB mode with the shared key K_{CID}. Because the K_{CID} is shared only between the customer and the SMS server, the message authentication is approved. Then the *CID* and the encryption of K_s are attached to the beginning and ending of the ciphertext of the message encrypted with K_s, respectively. Finally, the whole message is sent to the SMS server as an SMS message.

Once the SMS server receives the SMS message, it indexes its database against the received *CID* to get the corresponding IV_{CID} and K_{CID}. This key together with the AES decryption algorithm is used to recover the k_s. Next, K_s, IV_{CID}, and AES decryption algorithm in CBC mode are used to recover the encrypted message. Then, the *MAC1* of the received message is compared with the received *MAC*, where *MAC1 is* obtained in the same way as *MAC* obtained in Figure 10a. If they are the same, the message integrity is approved. Finally, the freshness of the received message is assured by testing the *Nonce* against the old nonces within the timestamp ST. This whole process is depicted in Figure 10b.

5. PROVIDED SECURITY DISCUSSION

This section is dedicated to present how the SHPS provides confidentiality, integrity, authentication, and protection against replay attacks. The system achieves the confidentiality by randomly generating a session key k_s together with an initial vector *IV* to each message and uses them to encrypt the message using AES algorithm in CBC mode. The AES algorithm uses a key with128 bits and its security is approved to be currently secure. Next, the k_s is encrypted with the K_{CID} and then attached to the message to enable the recipient to recover the message. Since only both the customer and the SMS server know the shared key K_{CID}, this approved the authentication. Also the k_s, *IV*, and *Nonce* are unique for each message. Thus, the produces ciphertext will be unique each time even if the same message request is performed again. This strengthens the security of the underlying system (i.e., SHPS).

The integrity is achieved through generating a unique *MAC* related to the generated message at the sender while the recipient verifies this *MAC*. This is also approved authentication since the *MAC* is generated with the shared secret key K_{CID} which is only known to the SMS server and the customer. In other words, the *MAC* cannot be generated by any other entity except the customer and it cannot be verified by any other entity

except the SMS server. Also, the session key, k_s, cannot be recovered by any other entity except the SMS server. Therefore, both the integrity and authentication are achieved.

Also the proposed system keeps track of each transaction that takes place and stores the related information into a dedicated database. In addition, the action performed on the bank accounts according to a system transaction is considered as evidence on both parties involved in the transaction. Thus, the system satisfies the non-repudiation of the involved parties.

Finally, the proposed system SHPS protects against replay attacks through providing a unique nonce and a time stamp for each message. This means that no two messages have the same nonce and time stamp. Therefore, if a recipient receives two messages with the same nonce or time stamp, this is an indication of a replayed message. Accordingly, we approved that the proposed SHPS achieve the required security properties.

6. Conclusion

During the course of this paper, we proposed and designed a new payment system called SHPS which allows people to make their payments from anywhere at any time in a secure environment. SHPS overcomes the shortcomings associated with most of the current SMS-based mobile payment systems. SHPS architecture is composed of three main components: *customer*, *bank* and *provider*. The formats of messages conveyed among the system components were introduced. Also, the services provided by the SHPS were presented. In addition, algorithms for key generation, distribution, MAC generation and verification were developed. Furthermore, the security properties (i.e., confidentiality, integrity, authentication, non-repudiation, and protection against replay attacks) provided by the SHPS were discussed and approved. Finally, the proposed system SHPS is designed to be applied in the KSA.

In future, the SHPS system will be adapted to accommodate customers and providers that have bank accounts in different banks. Also SHPS can be adapted as a voting system to securely allow people to make their votes through SMS messages from anywhere during the election time. Furthermore, it can be adapted to be used as a secure fault reporting system.

REFERENCES

[1] Payment Systems Regulator, "A new regulatory framework for payment systems in the UK," PUB REF: PSR CP14/1, 2014: www.psr.org.uk. Last retrieved on January 1, 2015.

[2] Treasury Alliance Group, "Fundamentals of Payment Systems," 2014: www.treasuryalliance.com.

[3] The Federal Reserve Banks, "Payment System Improvement -Public Consultation Paper," September 2013: Last retrieved in January 2015. https://fedpaymentsimprovement.org/.

[4] G. Carat, "Mobile payments: Alternative platforms and players." *IPTS Report*, Vol. 49, November 2000.

[5] P. Y. K. Chau and S. Poon, "Octopus: an e-cash payment system success story," *Communications of the ACM*, 46 (9):129–133, 2003.

[6] A. M. Moore, "Replacing Cash with Convenience: The Promise of RFID Payments," *Celent communication*, September 2003. Last accessed in Dec. 2014: www.banktech.com/showArticle.jhtml?articleID=14700510

[7] E.W. Ngai, Karen K. Moon, Frederick J. Riggins, and Candace Y. Yi, "RFID research: An academic literature review and future research directions," International Journal of Production Economics, Vol. 112 (2), PP. 510–520, April 2008.

[8] H. Harb, H. Farahat, and M. Ezz, "SecureSMSPay: Secure SMS Mobile Payment Model," Proceedings of the 2nd International Conference on Anti-counterfeiting, Security and Identification, Guiyang, pp. 11-17, 2008.

[9] M. Toorani and A B. Shirazi, "SSMS - A Secure SMS Messaging Protocol for the M-Payment Systems," Proceedings of the 13th IEEE Symposium on Computers and Communications, Marrakech, pp. 700-705, 2008.

[10] S. Kungpisdan, B. Srinivasan, and PH Le, Lightweight Mobile Credit-card Payment Protocol, Lecture Notes in Computer Science, Vol. 2904, pp. 295-308, 2003.

[11] S. Duangphasuk, M.Warasart, and S. Kungpisdan, "Design and Accountability Analysis of a Secure SMSBased Mobile Payment protocol," Proceedings of the 8th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, pp. 442–445, 2011.

[12] M. R. Hashemi and E. Soroush, "A Secure m-Payment Protocol for Mobile Devices," In Proceedings of the Canadian Conference on Electrical and Computer Engineering (CCECE'06), PP. 294-297, Ottawa, Ont, 2006.

[13] Pavan Soni, "M-Payment Between Banks Using SMS," In the Proceedings of the IEEE, Vol. 98 (6), pp. 903-905, June 2010, ISSN: 0018-9219.

[14] Neetesh Saxena and Narendra Chaudhari, "SecureSMS: A secure SMS protocol for VAS and other applications," The Journal of Systems and Software, Vol. 90, pp 138- 150, 2014.

[15] JieWang, "Computer Network Security: Theory and Practice." Springer Berlin Heidelberg, New York, 2009.

[16] D. Eastlake and P. Jones "US Secure Hash Algorithm 1," Network working group RFC 3174, Sep. 2001.

[17] R. L. Rivest, A. Shamir, and L. Adleman, "A Method for obtaining Digital Signatures and Public-key Cryptosystems," Communications of the ACM, Vol. 21 (2), pp 120 – 126, New York, NY, USA, 1978.

[18] Steven J. Murdoch, Ross Anderson, "Security Protocols and Evidence: Where Many Payment Systems Fail," Financial Cryptography and Data Security, Barbados, March 2014.

[19] Ronald L. Rivest, The MD5 Message Digest Algorithm, Internet RFC 1321, April 1992.

Aly M. El-Semary received his B.S. degree in Systems and Computer Engineering from Al-Azhar University, Cairo, Egypt in 1992, and M.S. and Ph.D. degrees in Computer Science from Tulsa University, USA in 2001 and 2004, respectively. He works for the Department of Systems and Computer Engineering, Faculty of Engineering, Al-Azhar University, where he is currently an associate Professor. However, he is currently working as a visitor for Computer Science and Engineering College, Taibah University, Saudi Arabia. His current interests include network and computer security, sensor networks, fuzzy logic, data-mining, and neural networks. He is a member of IEEE and the author of several research papers published in the most reputable journals and conferences. He is also a working group member of IEEE 1903 standard on Next-Generation Service Overlay Networks.

Abdullah A. AlAnnoor is an undergraduate student at Taibah University. He is expected to be graduated in spring of 2015. *AlAnnoor* has the Cisco CCNA certificate and Cambridge Information Technology. He is experience in Java programming language and in the development of Android applications. He worked as summer training for the Deanship of Information Technology at Taibah University. ALAlnnoor together with his group won the first position in the student preparation conference on the university level and in the general student conference in 2015 on all Saudi Universities level, respectively.

Ahmad A. Alshammari is an undergraduate student at Taibah University. He is expected to be graduated in spring of 2015. He also has the Cisco CCNA certificate. He is experience in Java programming language and in the development of Android applications. He worked as summer training for the Deanship of Information Technology at Taibah University. *Alshammari* together with the rest of his group won the first position in the student preparation conference on the university level and in the general student conference in 2015 on all Saudi Universities level, respectively.

Abdulaziz E. Aljohani is an undergraduate student at Taibah University. He is expected to be graduated in spring of 2015. He also has the Cisco CCNA certificate. He is experience in Java programming language and database development applications He worked as summer training for the Deanship of Information Technology at Taibah University. *Aljohani* along with his group won the first position in the student preparation conference on the university level and in the general student conference in 2015 on all Saudi Universities level, respectively.