SOMP: An SMS-based Operator-assisted Mobile Payment Protocol

Supakorn Kungpisdan[1] and Maykin Warasart[2]
[1]Faculty of Information Science and Technology, Mahanakorn University of Technology
140 Cheumsampan Road, Nong Chok, Bangkok 10530, Thailand, Tel. 0-2988-3655, Email: supakorn@mut.ac.th
[2]Faculty of Information Science and Technology, Mahanakorn University of Technology
140 Cheumsampan Road, Nong Chok, Bangkok 10530, Thailand, Tel. 0-2988-3655, Email: maeklong@hotmail.com

Abstract

Mobile Payment allows users to purchase products or services on the move. Nowadays, most of mobile payment transactions are made through Short Message Service (or SMS). A number of SMS-based mobile payment protocols have been proposed but still lack of necessary security properties. This paper introduces a new SMS-based operator-assisted mobile payment protocol called SOMP. The proposed protocol offers clients the ability to perform payment transactions directly to the mobile operator itself or to merchants through the mobile operator. Our protocol not only satisfies necessary transaction security properties, but it is also simple and compatible with existing SMS infrastructure.

Keywords: Mobile payment, SMS payment, mobile commerce, payment protocols, cryptographic protocols

1. Introduction

Mobile device capability nowadays is far beyond making phone calls. The current generation of mobile telecommunications offers the ability to transmit data through mobile communications networks e.g. GPRS (General Packet Radio Service), EDGE (Enhanced Data rates for GSM Evolution) or HSPDA (High-Speed Downlink Packet Access). People can connect to the Internet and run several network applications on their mobile devices. Although several forms of data transmission are available due to the high speed data transmission, the most popular data transmission is still Short Message Service (or SMS for short) due to its low cost, and this feature is fundamentally embedded in most of low cost mobile devices. SMS nowadays has been used for several purposes. For example, a mobile operator sends SMS text messages to its subscribers as a part of advertising campaigns. People send SMS text messages to request for services.

Currently, SMS is now used for payment transactions. People purchase ringtones, songs, or pictures from their corresponding mobile operator. Such kinds of transactions are known as *Mobile Payment*. Mobile Payment is defined as a payment transaction between engaging parties whereas a payer transfers an amount of money to a payee, and receives a product or service from the payee. Mobile payment is not just an alternative of electronic payment. The nature of wireless environments leads to a number of emerging issues regarding transaction security and performance to mobile payment. On one hand, the data transmitted over a wireless network is easily eavesdropped. Although a GSM network provides an encryption of traffic transmission between a mobile device and a base station, the relying cryptographic techniques, A5/1 and A5/2, are too weak and its vulnerability is reported [1]. Thus, it can be argued that the security of GSM network at data-link layer is not sufficient; it is necessary to have a secure communications channel established over the wireless network at the higher layer. In our perspective, application layer security can be practically implemented. This can be achieved primarily by deploying cryptographic operations.

To secure mobile payment based on SMS text messaging, a number of mobile payment protocols have been proposed [1, 2, 3]. Toorani *et al.* [1] proposed SSMS, a secure SMS messaging protocol, based on elliptic-curve cryptography that satisfies confidentiality, integrity, authentication, and non-repudiation. Moreover, the protocol provides the ability to verify 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.

Hashemi *et al.* [3] proposed a framework for secure mobile payment based on SMS that describes well the interaction between engaging parties including SMS gateway and SMSC (or Short Message Service Center) and also describe an overview of various types of payment schemes that are suitable for SMS-based payment. The transactions between engaging parties are secure by using AES (Advanced Encryption Standard), a symmetric cryptographic algorithm. However, the shared keys between the client and the bank is distributed only when the client registers for the service for the first time, but no session key update were discussed in the paper.

Harb *et al.* [2] proposed SecureSMSPay, a payment system between a payer and a payee. The payment transfer is made from the payer's bank to the payee's bank via a payment gateway. However, the system requires the payment gateway to know both payer and payee's mobile phone numbers. Moreover, the security of the system is based on symmetric cryptography which requires a shared key between engaging parties. The system changes session keys based on the hash value of cyclic shift of current session key. This is vulnerable to attacks.

In this paper, we introduce a new secure mobile payment protocol based on SMS messaging called an **S**MS-based **O**perator-assisted **M**obile **P**ayment Protocol (SOMP). Our protocol is lightweight as only symmetric cryptographic operations and hash functions are used. The proposed protocol allows a client to make payment to a mobile operator who offers products or services or to a merchant through the mobile operator. This makes the proposed protocol practical as a real-world application. Moreover, the proposed protocol satisfies necessary transaction security properties, that is, message confidentiality, message integrity, and message authentication. Moreover, we apply a limited-used session key generation and distribution technique [8] to prevent the reuse of session keys during the transaction. Furthermore, the proposed protocol is compatible with existing SMS messaging infrastructure.

This paper is organized as follows. Section 2 discusses related works and necessary backgrounds for this paper. In section 3, we describe the proposed protocol in details. Section 4 shows security analysis of the proposed protocol. In section 5, we conclude our work.

2. Related Works

2.1 Mobile Payment Overview

According to [14], A general payment system is composed 5 engaging parties: *client*, *merchant*, *payment gateway*, *issuer* (the client's financial institution), and *acquirer* (the merchant's financial institution). The payment gateway operates on behalf of the issuer and the acquirer on the Internet side, whereas the payment clearings are performed directly between the issuer and the acquirer over a banking private network. There are 3 primitive transactions: *Payment Ordering*, *Debit*, and *Credit*.

Payment is the interaction between the client and the merchant in that the client requests to purchase goods or services with the merchant. Also, it is the interaction that the merchant sends the payment receipt to the client in return. *Debit* is made by the client in order to request the payment gateway (on behalf of the issuer) to deduct the requested amount from the client's account. Also, the payment gateway notifies the client that the requested amount has been deducted from the client's account. *Credit* is made by the merchant in order to request the payment gateway (on behalf of the acquirer) to transfer the requested amount to the merchant's account. Also, it is the transaction that the payment gateway (on behalf of the acquirer) notifies the merchant that the requested amount has been transferred or committed to be transferred. Payment transactions in several payment protocols [11, 12, 13] are based on the following steps:

C → M: *Payment (Request), Debit (Request)*
M → PG: *Debit (Request), Credit (Request)*
PG → M: *Credit (Response), Debit (Response)*
M → C: *Payment (Response), Debit (Response)*

Where *{C, M, PG}* stands for the set of client, merchant, and payment gateway, respectively. However, a number of payment protocols operate differently. That is, in some payment systems, the payment gateway acts as a centralized party where the transactions between the client and the merchant must be processed through it. An obvious example for this kind of payment systems is Internet banking systems. The following process shows the payment in terms of the above primitive transactions:

C → PG: *Payment (Request), Debit (Request)*
PG →M: *Payment (Request)*
M → PG: *Credit (Request)*
PG → M: *Credit (Response)*
M → PG: *Payment (Response)*
PG→ C: *Payment (Response), Debit (Response)*

It can be seen that the above transactions has to be performed through a middle party which is the payment gateway. It is believed that the above process fits well to the current SMS-based mobile payment system whereby a mobile operator acts as the payment gateway (and the issuer). That is, a client is subscribed to a mobile operator for services including making payments. The products or services include ringtones, songs, video clips, etc. The client is authorized to make phone calls including purchase products or services within certain credit limits. Then, the mobile operator transfers the purchased amount to the merchant.

2.2 Existing SMS-based Mobile Payment Protocols

In this section, we discuss advantages and limitations of a number of existing SMS-based mobile payment systems: Hard *et al.*'s approach, Toorani *et al.*'s approach, and Hashemi *et al.*'s approach, respectively.

Harb *et al.* [2] proposed SecureSMSPay, a payment system between a payer and a payee based on symmetric cryptography. According to SecureSMSPay, there are 5 engaging parties: payee, payer, payee's bank, payer's bank, and payment gateway. A payee established an account with the payee's bank, whereas a payer establishes one with the payer's bank. A payment gateway acts as an intermediate party forwarding requested between banks. The payment transfer is made from the payer's bank to the payee's bank via a payment gateway. The following steps show details of SecureSMSPay:

Payee → Payee's Bank: $\{Mobile_{Payer}, i, OI\}_{Xi}$
Payee's Bank → Payee: $Mobile_{Payer}, OI, Mobile_{Payee}$
PG → Payer's Bank: *MobilePayer, OI,* $Mobile_{Payee}$
PG: router transaction to Payer's Bank based on $Mobile_{Payer}$
Payer's Bank → Payer: $\{Mobile_{Payee}, j, OI\}_{Yj}$
Get payer confirmation (Y/N)
Payer → Payer's Bank: $\{Mobile_{Payee}, Status, TID\}_{Yj+1}$
Payer's Bank → PG: $Mobile_{Payee}, Status, TID$
PG → Payee's Bank: *Status, TID*
Payee's Bank → Payee: $\{Status, TID\}_{Xi+1}$

Note that $\{M\}_K$ stands for a message M symmetrically encrypted with a key *K*. X_i, $i = 1, ..., n$, are symmetric keys shared between the payee and its bank, and Y_j, $j = 1, ..., n$, are symmetric keys shared between the payee and its bank. The generation of these shared keys based on Kungpisdan *et al.*'s technique [14] by applying hash function to one-bit-left shift to the current value of symmetric key in order to produce a new key.

It can be argued that the above system has some flaws. That is, some messages are sent in cleartext. For example, the payee's mobile number $Mobile_{Payee}$ and *Status* can be modified by an attacker. Moreover, the security of the system is based on symmetric cryptography which requires a shared key between engaging parties. The system changes session keys based on the hash value of cyclic shift of current session key. It can be noted that a secret key is produced from hash function that produces a fixed-length output. Applying a hash function to the 1-bit-left shift of the current key does not significantly increase security because the attacker will finally be successful on performing brute-force attack to the current key. For example, a secret key is generated from SHA-1. An output of SHA-1 is 128-bit long. The success rate on performing brute-force attack to a 128-bit message is $(1/2^{128})/2$. This is considered not impossible. Thus, this protocol is vulnerable to attacks.

Toorani *et al.* [1] proposed SSMS, a secure SMS messaging protocol, based on elliptic-curve cryptography that satisfies confidentiality, integrity, authentication, and non-repudiation. Moreover, the protocol provides the ability to verify 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.

Hashemi *et al.* [3] proposed a framework for secure mobile payment based on SMS that describes well the interaction between engaging parties including SMS gateway and SMSC (or Short Message Service Center) and also describe an overview of various types of payment schemes that are suitable for SMS-based payment. The transactions between engaging parties are secure by using AES (Advanced Encryption Standard), a symmetric cryptographic algorithm. However, the shared keys between the client and the bank is distributed only when the client registers for the service for the first time, but no session key update were discussed in the paper.

2.3 Limited-Used Session Key Generation and Distribution

Session key generation and distribution is one of the most widely discussed topics in symmetric cryptography. This is because a secret key needs to be shared between engaging parties in a secure manner. A number of session key generation and distribution techniques have been proposed [8, 9, 10, 13, 14, 15]. Among these techniques, they can be classified in two types: *online* and *offline* techniques. On one hand, online session key generation and distribution techniques require a new session key to be transmitted over the network. Although it is transmitted in an encrypted format, it is possible that the key can be compromised. On the other hand, in an offline session key generation and distribution technique, a new session key is not necessary to be transmitted in the network. Thus an attacker is not able to capture the session key on the wire.

According to the above discussion, an offline session key generation and distribution was chosen to secure our proposed secure instant messaging protocol. Several offline session key generation techniques have been proposed [8, 9, 10, 13, 14, 15]. Among them, Kungpisdan *et al.* [8] introduced a session key generation technique that not only it is secure against key compromise attacks, but it can also operate purely offline. According to this technique, the longer the technique is used, the more secure the internet transaction will be. Kungpisdan *et al.* argued that this technique can be applied to any kind of transactions.

Kungpisdan *et al.*'s Approach

Assume that Alice and Bob share $\{K_{AB}, DK, m\}$, where K_{AB} is a long-term key, DK is called a *distributed* key, and m is a random number. m is used to specify the number of keys that will be generated. m also varies randomly among different pairs of parties. $conc(M_1, M_2, M_3)$ represents the concatenation of the message M_1, M_2, and M_3, respectively. Then, the key generation process is shown in Figure 1.

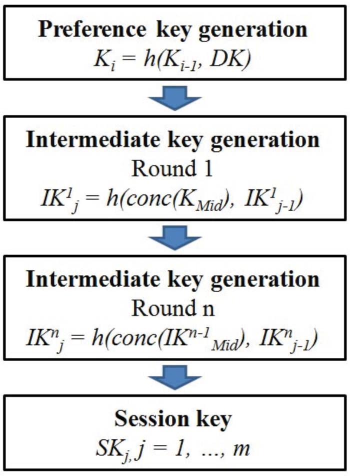
Preference key generation
$K_i = h(K_{i-1}, DK)$
Intermediate key generation
Round 1
$IK^1_j = h(conc(K_{Mid}), IK^1_{j-1})$
Intermediate key generation
Round n
$IK^n_j = h(conc(IK^{n-1}_{Mid}), IK^n_{j-1})$
Session key
$SK_j, j = 1, \ldots, m$

Figure 1: Session Key Generation

After sharing $\{K_{AB}, DK, m\}$, Alice and Bob generate a set of *preference keys* K_i, where $i = 1, \ldots, m$, as follows: $K_i = h(K_{i-1}, DK)$, where $K_0 = K_{AB}$. The set of K_i will be used as a source to regenerate session keys if needed. After generating the set of K_i, K_{AB} and DK can be removed from the system.

Then both Alice and Bob create sets of *intermediate keys* in order to increase the difficulty for cryptanalysis. In other words, it increases difficulty to trace back to the preference key if the session key is compromised. In each round, a new set of intermediate keys is created. The higher number of round is performed, the greater security the system is. The intermediate key generation is performed as follows: $IK^x_j = h(conc(IK^{x-1}_{Mid}), IK^x_{j-1})$, where x specifies the round number, j specifies the number of intermediate keys that is generated, $j = 1, \ldots, m$. IK^{x-1}_{Mid} stands for the set of $\{IK^{x-1}_{Mid1}, IK^{x-1}_{Mid2}, IK^{x-1}_{Mid3}\}$. $IK^x_{Mid1} = mid(IK^x_1, IK^x_{rm})$ and rm is the remaining number of intermediate keys in the set of IK^x_j. $IK^x_{Mid2} = mid(IK^x_{Mid1}, IK^x_{rm})$. $IK^x_{Mid3} = mid(IK^x_1, IK^x_{Mid2})$. $IK^1_{Mid1} = K_{Mid1}$, $IK^1_{Mid1} = K_{Mid2}$, and $IK^1_{Mid1} = K_{Mid3}$. The generation of K_{Mid1}, K_{Mid2}, and K_{Mid3} is the same as that of IK^x_{Mid1}, IK^x_{Mid2}, IK^x_{Mid3}, respectively. $IK^x_{j-1} = \phi$. The output of the last round of intermediate key generation is considered as session keys SK_j, where $j = 1, \ldots, m$, which is shown below: $IK^n_1 = SK_1$, $IK^n_2 = SK_2, \ldots, IK^n_m = SK_m$. Alice and Bob then can use SK_j as a credential to secure transactions e.g. as an encryption key or as an input to message authentication code.

It can be clearly seen that the session key was generated purely offline. Each engaging party can create a set of session keys used to secure communications between them without the needs to transfer credentials over the network. As a new session key is not transferred over the network. Thus, it will never be intercepted. Thus, this technique is expected to increase security of symmetric-key cryptosystems including the proposed SMS-based mobile payment protocol.

3. The Proposed Mobile Payment Protocol

3.1 Overview of the Proposed Protocol

In this section, we propose a new SMS-based mobile payment protocol. The following assumptions are made:

- The system is composed of 3 engaging parties: a client *C*, a merchant *M*, and a mobile operator *O*. *C* is using a mobile device installed with the proposed SMS payment software. M is registered as a merchant under *O*. *O* itself setups a server call SMS Payment Server (SPS) to provide payment services to both *C* and *M*.
- *C* establishes an account with *O* for both phone and data usage. *C* is billed by *O* at the end of each month.
- $\{A, B\}$ is the set of communicating users, whereas *S* denotes an instant messaging server.
- ID_A is the identity of *A*.
- $\{DK_{AB}, K_{AB}, m_{AB}\}$ is the of key distribution parameters of session key generation and distribution. Note that these parameters are shared between a party *A* and a party *B*.
- SK_{ABj}, where $j = 1, ..., m$, stand for session keys shared between the party *A* and the party *B*.
- *n* is a nonce to prevent replay.
- $\{m\}_K$ is a symmetrically encrypted message of a message *m* with a key *K*.
- $h(m)$ is a hash value of a message *m*.
- $h(m, K)$ is message authentication code (MAC) of a message *m* and a key *K*.

3.2 Client Registration

Each mobile device is installed with SMS payment software based on the proposed protocol. Once the software has been downloaded to the client, the client is required to logon to the mobile operator's SPS in order to register to the system. The registration can be performed over a secure channel e.g. TLS (Transport Layer Security). Note that the specific version of TLS which is deployed to secure wireless communications is called WTLS (Wireless Transaction Layer Security). The purpose of registration process is to share necessary secrets $\{K_{CO}, DK_{CO}, m_{CO}\}$ between the client and the mobile operator. Alternatively, each client's mobile device may be installed with SIM Application Toolkit (SAT) which contains the above secrets shared with the mobile operator. Note that after sharing $\{K_{CO}, DK_{CO}, m_{CO}\}$, both the client and the mobile operator can create a set of session keys SK_{COj}, where $j = 1, ..., m$, by using the session key generation technique presented in section 2.3.

3.3 Direct Payment to Mobile Operator

In this section, we introduce a protocol what is suitable for making payment between a client and a mobile operator through SMS messaging. Assume that the mobile operator *O* offers products and services to its clients, for example, ringtones, music, and application downloads. Then the client is charged from what he/she purchases into his/her account that will be billed at the end of the month.

The proposed protocol allows both prepaid and postpaid payment transactions to be performed as shown in details below.

To perform a prepaid payment transaction, a client is required to purchase a cash card that is normally available in convenient store. Then the client tops up the purchased amount on to his/her account. Then, the client performs a payment transaction as follows:

Purchase Credit Request

After purchasing a top-up cash card, the client *C* opens the client software from his/her device, fill in the necessary information and sends the following to the mobile operator *O*:

C → O: $ID_C, T_1, h(SN, CL_T, T_1, SK_{COj}), SN$

Where,

- *SN* is a serial number of a prepaid cash card purchased offline. Note that *SN* infers the credit limit CL_T that the client is allowed to purchase products or services from *O*.
- ID_C = the client's identity. It can be inferred from the client's mobile phone number.
- T_1 = Timestamp when requesting the purchase credit

From the above message, the client sends SN together with the requested credit limit CL_T to *O*. *O* can infer CL_T from *SN*. Note that, an attack cannot modify SN even though SN is transmitted in cleartext as it is key-hashed in $h(SN, CL_T, T_1, SK_{COj})$ which is a message MACed with SK_{COj} shared between *C* and *O*. After receiving the request from *C*, *O* adds the value CL_T to *C*'s account and sends the following message to *C*:

O → C: $T_1, T_2, h(CL_T, T_{S1}, T_2, SK_{COj+1})$

Where, T_2 stands for the timestamp when issuing the purchase credit to the client. From the above message, *O* acknowledges the increment of *C*'s credit by CL_T. *C* is currently ready to make a payment.

Making Payment

After browsing for products or services, *C* can make a payment by sending the following message to *O*:

C → O: $ID_C, \{T_P, OI\}_{SKcoj}, h(T_P, OI, ID_C, SK_{COj})$

Where

- T_P stands for the timestamp when making a payment request.
- $OI = \{TID, Price, OD\}$. *TID* stands for Transaction ID, *Price* is the price of product or service, and *OD* stands for order descriptions containing details of the product or service purchased.

After receiving the request, *O* checks the remaining credits of *C* and compares with the requested amount *Price*. If *C* has sufficient credits, *O* responses with *Yes* message back to *C*. If not, *O* responses with *No* attached to the message below:

O → C: $Yes/No, h(Yes/No, CL_{RM},$
$h(T_P, OI, ID_C, SK_{COj}), SK_{COj+1})$

Note that CL_{RM} is the remaining credits in *C*'s account. Note also that postpaid payment protocol can be performed in the same manner as the prepaid payment without the Purchase Credit Request protocol. This is because the client has already been authorized certain credit limit CL_T from the mobile operator.

3.4 An SMS-based Operator-Assisted Mobile Payment Protocol

Based on the payment model presented in section 2.1, in this section we introduce an SMS-based operator-assisted mobile payment protocol (or SOMP for short). In this protocol, the mobile operator assists a client while performing a payment transaction with a merchant.

The following assumptions are made for SOMP. The system is composed of client *C*, merchant *M*, and mobile operator *O*. The client and the merchant established the accounts with *O*. *C* has been authorized certain amount of credit limit from the mobile operator and the client will be billed each month by the mobile operator for any products or services purchased. The client-side application performs two main functions: product search and making payments. The merchant *M* is defined as a merchant who places products and services on a mobile portal operated by the mobile operator.

The details of SOMP are shown as follows:

1) After deciding to use the payment service, *C* establishes a WTLS session and shares $\{K_{CO}, DK_{CO}, m_{CO}\}$ with *O*. *C* and *O* then create a set of session keys K_{COj}, where $j = 1, ..., m$, by using the secure key generation technique presented in section 2.3.
2) *M* shared $\{K_{MO}, DK_{MO}, m_{MO}\}$ with *O*. Both of them create a set of session keys K_{MOj}, where $j = 1, ..., m$, by using a secure session key generation technique.
3) *C* opens the client program installed on his/her device to browse for products or services. Once the products or services are added to a cart, *C* performs a request to purchase product or service as follows:

   C → O: $ID_C, T, \{ID_M, OI, T, h(OI, K_{CMj})\}_{KCOj}$
   O → M: $\{OI, h(OI, K_{CMj}), h(OI, K_{COj+1}), T\}_{KMOj}$
   M → O: $\{Yes/No, h(Yes/No,OI, K_{CMj+1})\}_{KMOj+1}$
   O → C: $\{Yes/No, CL_{RM}, h(Yes/No, OI, K_{CMj+1}),$
   $h(OI, K_{MOj+1})\}_{KCOj+1}$

Where, *T* stands for the timestamp. Note that after *C* clicks on the checkout button, a new session is established between *C* and *M*. A set of secret keys $\{K_{CM}, DK_{CM}, m_{CM}\}$ is distributed between them. Then, both of them can create a set of session keys SK_{CMj}, where $j = 1, ..., m$, based on the offline session generation and distribution technique.

From the above messages, one can notice that *O* cannot create the first message by itself because the message contains $h(OI, K_{CMj})$ shared between *C* and *M*.

4. Discussions

4.1 Compatibility with Existing Infrastructure

The proposed protocol was designed with concerning about the compatibility with existing SMS infrastructure. That is, each SMS message allows only 160 characters (or 160 bytes). Thus, we try to reduce the size of messages by employing hash functions and message authentication code which reduce the size of the message to 20 bytes (with SHA-1). Moreover, the deployment of symmetric cryptographic operations produces small-size messages. According to our design, none of the messages in the proposed protocol has the size greater than 160 bytes.

Moreover, normally, applying cryptographic operations such as symmetric encryption or hash functions produces an output in binary format. Thus, to make each message compatible with the existing SMS infrastructure, a Binary-to-ASCII algorithm, such as BASE64, may be applied to each message before the transmission.

4.2 Security of Payment Transactions

One of the major concerns while performing mobile payment transaction is security. That is, a mobile payment protocol should satisfy necessary transaction security properties such as message confidentiality, message integrity, message authentication, and forward secrecy.

According to the proposed protocol, message confidentiality is ensured by applying symmetric encryption. That is, only the parties who possess shared keys are able to decrypt the message. The message integrity is ensured by applying message authentication code to ensure that an attacker will not be able to modify the message without being detected by the receiver. Only the party who knows the appropriate MAC key is able to verify the integrity of the message. The message authentication is ensured by applying both symmetric encryption has message authentication to each message.

Furthermore, in this paper, we concern about the security of session keys. That is, each session key should not be reused. According to the proposed protocol, a limited-used session key generation and distributed technique was applied to ensure that each message contains only fresh session keys and such session keys will not be transmitted between engaging parties before the transmission of the message. This reduces the chance to be attacked.

It could also be noted that the proposed protocol also satisfies forward secrecy. That means, the security of the system still persists although one of the session keys is compromised. Consider the case that an attacker intercepts an encrypted message and successfully derives a session key SK_{CM1} shared between *C* and *M*. The attacker cannot use SKCM1 for decrypting any other messages as the key is used only once. Moreover, based on the technique proposed in [8], the attacker cannot derive SK_{CM2} from possessing SK_{CM1}.

5. Conclusions

In this paper, we found that Short Message Service (SMS) has become the most common communications channel between users and users and between users and mobile operators. We found that making payment using SMS seems to be a promising application, but existing approaches to secure SMS-based payment transactions are not sufficient as they lack of both security properties and performance. In this paper, we introduced a new SMS-based mobile payment protocol called SOMP that allows users to purchase products and services from their mobile operator and merchants. The proposed protocol not only satisfies necessary security properties, but it is also compatible with existing SMS infrastructure.

As our future works, we aim to build a prototype of the proposed protocol to prove its practicability as a real-world application.

References

[1] M. Toorani and A. A. B. Shirazi, SSMS – A Secure SMS Messaging Protocol for the M-Payment Systems, Proceedings of the 13th IEEE Symposium on Computers and Communications (ISCC'08), Marrakech, July 6-9, 2008, pp. 700-705.

[2] 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 2008, Guiyang, Aug 20-23, 2008, pp. 11-17.

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

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

[5] S. Kungpisdan, Accountability in Centralized Payment Environments, Proceedings of the 9th International Symposium on Communications and Information Technology 2009, Sept 28-30, 2009, Incheon, pp. 1022-1027.

[6] S. Kungpisdan and T. Thai-udom, Securing Micropayment Transactions Over Session Initiation Protocol, Proceedings of the 9th International Symposium on Communications and Information Technology 2009, Sept 28-30, 2009, Incheon, pp. 187-192.

[7] X. Wu, O. Dandash, and P. D. Le, The Design and Implementation of A Smartphone Payment System Based on Limited-used Key Generation Scheme, Journal of Theoretical and Applied Electronic Commerce Research, Vol. 1(2), Aug 2006, pp. 1-11.

[8] S. Kungpisdan and S. Metheekul, A Secure Offline Key Generation With Protection Against Key Compromise, Proceedings of the 13th World Multi-conference on Systemics, Cybernetics, and Informatics 2009, Orlando, USA.

[9] O. Dandash *et al.*, Fraudulent Internet Banking Payments Prevention using Dynamic Key, Journal of Networks, Vol.3(1), Academy Publisher, pp. 25-34, 2008.

[10] S. Kungpisdan, P.D. Le, and B. Srinivasan, "A Limited-Used Key Generation Scheme for Internet Transactions", Lecture Notes in Computer Science*,* Vol. 3325, 2005.

[11] M. Bellare, J. A. Garay, R. Hauser, A. Herzberg, H. Krawczyk, M. Steiner, G. Tsudik, E. V. Herreweghen, and M. Waidner, Design, "Implementation, and Deployment of the *i*KP Secure Electronic Payment System", *IEEE Journal of Selected Areas in Communications*, 2000.

[12] Mastercard and Visa, "SET Protocol Specifications", 1997.
http://www.setco.org/set_specifications.html

[13] Li, Y. and Zhang, X., 2004. A Security-enhanced One-time Payment Scheme for Credit Card. *Proc. of the Int'l Workshop on Research Issues on Data Engineering: Web Services for E-Commerce and E-Government Applications*,

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

[15] A. D. Rubin and R.N. Wright, Off-line Generation of Limited-Use Credit Card Numbers, Lecture Notes in Computer Science, Vol. 2339, 2002, pp. 196-209.