東海大學

資訊工程研究所

碩士論文

指導教授: 呂芳懌博士

植基於信用卡交易的一個安全移動電子商務系統

A secure m-commerce system based on credit card transaction

研究生: 王聖貿

中華民國一零四年六月

東海大學碩士學位論文考試審定書

東海大學資訊工程學系 研究所

研究生 王 聖 貿 所提之論文

植基於信用卡交易的安全移動電子商務系統

經本委員會審查，符合碩士學位論文標準。

學位考試委員會
召 集 人 羅濟舟 簽章

委 員 陳金鈴

黃宜孝

指 導 教 授 [illegible] 簽章

中華民國 104 年 6 月 29 日

摘　要

如今的無線網路購物需求正在不斷增加。但是信用卡詐騙的問題也很嚴重，現行的 SET 和 SSL 機制都有自己的問題。為了提高無線網絡購物的安全性，在本文中，我們提出了一個安全的移動電子商務計劃，稱為安全行動購物系統（簡稱 SMCS），用戶可以創建一個安全的信用卡交易進行網上購物。基本上，SMCS 協調交易系統的現金流和信用卡機構，有效防止各種不同的攻擊，避免信息洩露。所提出的系統還採用了數據鏈結核心（簡稱 *DCC*）來連接發卡銀行和消費者的無線通信在交易開始前，以顯著提高我們的移動購物環境的安全級別。理論分析表明，該 SMCS 比 SET 和 SSL 的更安全。性能分析則顯示 SMCS 確實是可行的行動購物系統。

關鍵字: 行動電子商務，二進制加法器，數據鏈結核心，傳輸層安全協議，安全電子交易標準

Abstract

Nowadays the demands for wireless Internet shopping are increasing. But credit card fraud has been serious, and SET and SSL have their own problems. To enhance the security of online shopping, in this paper, we propose a secure m-commerce scheme, called the Secure M-Commerce System (SMCS for short), with which users can create a secure-card transaction for Internet shopping. Basically, the SMCS coordinates the cash flow of a trading system and its credit card entities to effectively protect the issued transactions against different attacks and avoid information leakage. The proposed system also employs a Data Connection Core (*DCC* for short) to link the card-issuing bank and consumers before their wireless communication starts so as to significantly improve the security level of our m-commerce environment. Theoretical analysis shows that the SMCS is more secure than SET and SSL. The performance analysis indicates that the SMCS is indeed a feasible m-commerce system.

Keywords: M-Commerce, Binary adder, Data connection core, Secure Sockets Layer, Secure Electronic Transaction

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recently, the convenience and security of wireless communication have been greatly improved [1]. Many people enjoy online shopping with their credit cards. But due to the infrastructure of a wireless system, the transactions issued are created via wireless. On the other hand, credit card fraud nowadays is serious [2] [3], which significantly reduces online shopping attraction for some people. Also, owing to vigorous development of wireless networks, current mobile devices, such as mobile phones, tablet PCs and laptops, have provided users with diverse features and services, which have colored our everyday life and gradually changed people's shopping habits. Generally, a secure credit-card mechanism for m-commerce should securely protect the corresponding transactions and personal information. At present, when shopping in a wireless environment, e.g., to pay something by using the Secure Sockets Layer (SSL), one must send the card number, expiration date and other information to the merchant. In fact, SSL can ensure peer-to-peer delivery safety, but it cannot confirm the identities of the underlying users [4] [5].

To solve this problem, the Card network organizations Visa and MasterCard put forward an electronic payment system specification for Secure Electronic Transaction (SET) [6]. However, SET has its own problem, e.g., a consumers needs to apply for a certificate [7]. That means on user side, the corresponding information of the credit card must be stored in a hard disk. Also, to improve its security level, SET takes a long time to calculate complicated asymmetric encryption and

decryption key [8] [9], thus giving users an inconvenient m-commerce experience. Today, the increasing demands for m-commerce motivate us to construct a safe and convenient m-commerce mechanism. Therefore, in this study, we propose a secure m-commerce scheme, named the Secure M-Commerce System (SMCS for short) which coordinates the cash flow of a trading system and credit card entities to develop a safe and convenient m-commerce environment for users, without increasing extra restrictions and resources on the cash flow and credit card entities. Basically, we propose a credit-card dynamic authentication code to substitute for the credit card information so that the trading merchant cannot know the credit card number and its details. The SMCS also employs a Data Connection Core (*DCC* for short) to link the card-issuing bank and consumers before their wireless communication starts. Furthermore, the card-issuing bank authenticates the credit card's dynamic authentication code and merchant's dynamic authentication code rather than directly authenticating the credit card and merchant information. This can efficiently make sure the legitimation of the consumer and trading merchant so as to effectively increase the security level of the SMCS. Theoretical analysis shows that the SMCS is more secure than SET and SSL. The performance analysis indicates that the SMCS indeed a feasible m-commerce system.

The rest of this paper is organized as follows. Section 2 introduces background and related work of this study. Section 3 describes the proposed system. Performance and security are analyzed and discussed in Section 4. Section 5 concludes this paper and outlines our future studies.

Chapter 2

Background Review and Related Works

2.1 Background Review

2.1.1 Credit Card Transaction

Generally, the most important feature of a credit-card transaction is to transform the relationship on trading from "seller to buyer" into a series of contractual relations. Due to away from face-to-face purchase, the authorization and security will be the two major concerns. In such a transaction, after confirming the identity of a buyer, the seller receives guaranteed payment from the acquiring bank, and the acquiring bank also receives guaranteed payment from international organizations. The card-issuing bank then judges the authorization of the payment based on the payer' s up-to-date credit, and promises to fulfill the payment to the international organizations. Finally, the credit card holder (buyer) is obligated to settle the money with the card-issuing bank based on his/her credit-card contract. This seemingly complicated process, in fact, greatly simplifies the trading relationships between buyers and sellers, because the time difference between the payment and settlement system is no longer a problem, and the information flow and cash flow

are separated when the bank and the new contractual relationship intervene [10]. Also, the corresponding information flow can be recognized by the merchant immediately to authorize the transaction. Although the seller is requested to pay around 3% of total trading amount of price, this mechanism can greatly increase sale opportunities.

Meanwhile, the merchant is licensed with a message to confirm whether the transaction is completed, and authorization is only an instant of the information flow. Regarding the cash flow, for each day, all the network transactions from different participating member banks will be calculated later by the international organizations. After the member banks are recognized on the date of the network shopping, they will use the "real-time gross settlement system" to transfer the funds to the international organizations, and the international organizations transfer funds to the card-issuing bank. From this moment, you can say that the importance of the role a bank plays in this process is lower, since cash flow is really performed sometimes later after the information flow, and the purchase is completed after the accomplishment of information flow. VISA proves a thing "the information of money is sometimes more important than the money itself!"

Figure 1 illustrates an overview of the information flow of a credit-card transaction.

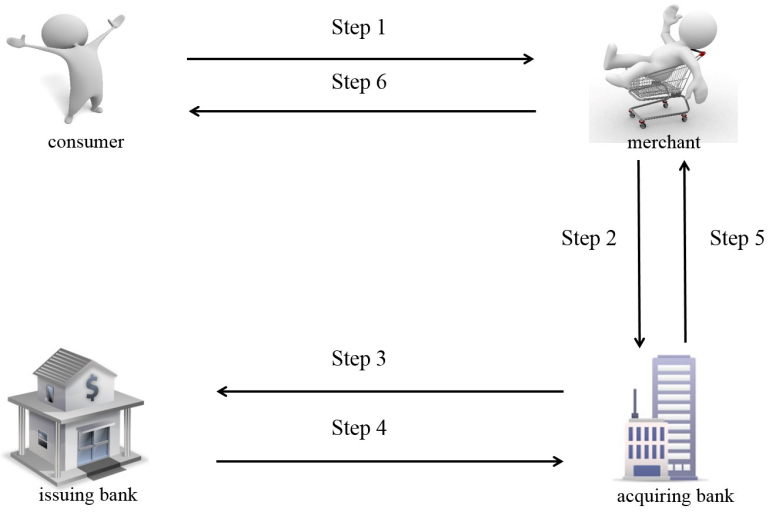

FIGURE 1: The information flow of a credit-card transaction.

Step 1: The consumer requests a purchase from the merchant.

Step 2: The merchant submits the request to the acquiring bank.

Step 3: The acquiring bank sends a request to the issuing bank to authorize the transaction.

Step 4: An authorization code is sent to the acquiring bank by the issuing bank by the issuing bank if a valid credit is available.

Step 5: The acquiring bank authorizes the transaction.

Step 6: The consumer receives the product.

Figure 2 illustrates an overview of the cash flow of a credit-card transaction.

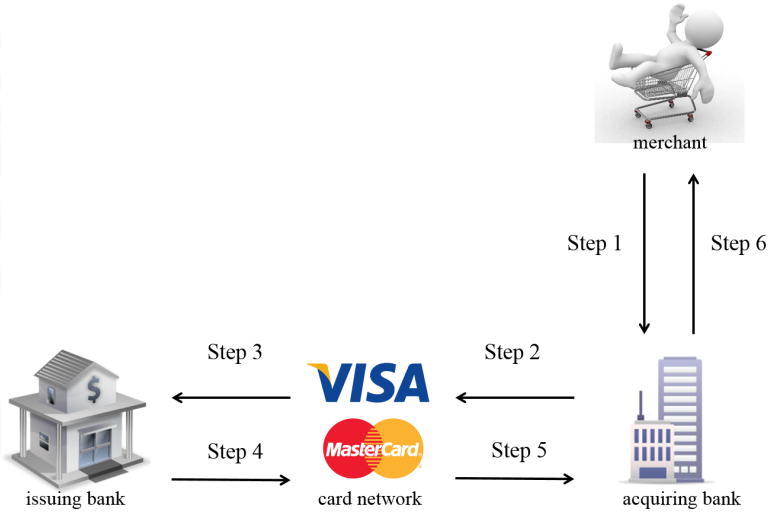

FIGURE 2: The cash flow of a credit-card transaction.

Step 1: The merchant sends a batch to the acquiring bank for payment P.

Step 2: The batch is delivered through the card network to request payment P from the issuing bank.

Step 3: The card network distributes a transaction to its corresponding issuing bank.

Step 4: The issuing bank subtracts its interchange fees F, which are shared with the card network, and transfers the remaining amount $R(= P - F)$to the international organization.

Step 5: The card network routes the remaining amount R to the acquiring bank.

Step 6: The acquiring bank subtracts its discount fee F' and pays the final remaining amount of payment $Q(= R - F')$ to the merchant.

2.1.2 Secure Sockets Layer (SSL)

SSL has two main features. The first is the use of a public-key and private-key mechanism to connect two sides of a network connection. With this mechanism, they can securely exchange encrypted messages with each other. The second is making use of the third party certification to enable both sides of the connection to confirm each other's information [11] [12].

SSL secures electronic transaction specification by using the consumer' s credit card number and expiration date or cardholder relevant information as the certification parameters, and transmits encrypted messages to the merchant. The merchant reuses the encrypted messages to request card-issuing bank for payment. The consumers prefer this way, because the system does not request users applying for an electronic wallet and a safety certification from the card-issuing bank.

But SSL has two shortcomings. The first is that the two sides of an SSL connection can only determine whether or not the other side is allowed to use the SSL mechanism. That means the consumer does not know who the merchant is, a legitimate merchant or a hacker. The merchant does not know the identity of the consumer, either, and also cannot confirm whether the consumer's credit card number is correct or not [13].

The second is that although SSL is convenient for consumers to perform Internet shopping through a wireless system, when SSL is invoked by a transaction, the card number and cardholder's related information can be clearly seen on the merchant side, thus possibly being unscrupulous businesses use. Besides, if the card number and other relevant information are stolen by hackers, they may be illegally

used for Internet shopping, causing the loss upon not only the cardholder, but also the merchant who would lose the unpaid products if the cardholder submits relevant evidences to deny this transaction. When SSL completes a transaction, the merchant cannot determine whether this transaction is completed before receiving the receipt from the funding or certified bank. The SSL handshake process on Credit card transaction has four stages [14] [15] [16]. In the first stage, consumer informs merchant what version of the SSL, an encryption-algorithm list and a compression-algorithm list that his/her terminal device supports. The merchant chooses the highest versions of SSL, an encryption algorithm and a compression algorithm for use. In the second stage, the merchant sends his/her own certificate and Diffie-Hellman's public key to the consumer. In the third stage, the consumer delivers its own certificate and Diffie-Hellman's public key to the merchant. With merchant's (consumer's) public key and consumer's (merchant's) own private key, consumer (merchant) can derive the Diffie-Hellman common secret key. In the fourth stage, a message is transmitted from acquiring bank to the merchant to prove that the key exchange and authentication process has been successfully completed.

2.1.3 Secure Electronic Transaction (SET)

SET was jointly developed by the VISA, MasterCard, IBM and other organizations [17]. Like SSL, it uses the public key and private key as the basis to secure message exchange process. However, SET requires that both consumer and merchant apply for SET' s certification and obtain the SET's electronic certification and software from card-issuing bank, and then use the software to complete a transaction online.

The greatest advantage of SET, unlike that in SSL, is that both trading sides of a connection can confirm each other's identity. In addition, SET can protect consumer's credit data, since the merchant only requires the consumer's SET credential before it can bill the card-issuing bank [18] [19].

With the SET mechanism, if a consumer wants to transact, his/her computer needs to install electronic wallet software [20], which like a real purse, is responsible for the storage of electronic cash. Before the transaction, the consumer has to first withdraw some amount of electronic cash from the bank. The bank then verifies the identity of the consumer, deducts the amount of money from the consumer's account, and deposits the amount of electronic cash to the consumer electronic wallet. After that, the consumer can purchase goods from manufacturers or shops. The above process is not very friendly to consumer since it is not an "enjoy-first-pay-later"mechanism. It has not achieved the stage of convenience for m-commerce anywhere [20].

2.1.4 Binary adder

The binary adder, denoted by $+_2$, is a new encryption method, which adds two binary numbers of the same length. When encrypting one of its two operands X (which is the plaintext) with the other operand K (which is the encryption key), it undergoes normal binary addition of X and K to generate ciphertext C, but ignores the overflow bit. To decrypt C, it compares C and K. If K is smaller, it binary subtracts K from C. Otherwise, it adds the two' s complement of K to C. Also, assume that both the two binary numbers X and K are m bits in length, then the probability p of recovering the values of (X, K) from intercepted $X+_2K$ on one trial is $p = 1/2^m$ [21] [22]. This encryption method provides a new choice for encrypting data (e.g., X) by using a key K. If we employ both the binary adder and exclusive-or operators to encrypt data, the security level of the underlying system will be greatly enhanced.

2.1.5 Data Connection Core (DCC)

From security viewpoint, in a wireless communication environment, there are two basic characteristics. (1) Wirelessly transmitted messages are insecure since hackers, the wireless system's legitimate staffs and users can receive the messages at the

same time. (2) A wireless system needs to confirm the identities of those presented correspondents. If the system and one of its users do not have any link before their wireless communication, the two entities at the beginning of their communication cannot create a secure channel for exchanging messages. Of course, the two entities also cannot mutually confirm each other' s identities by exchanging safe messages. This will cause serious problems, like credit card fraud or communication data leakage [21]. One of the methods to solve this problem is establishing an identity authentication mechanism between the two entities beforehand. We call the security mechanism the *DCC*, which is used to pre-link the wireless system and its users. For different security systems and communication mechanisms, *DCC* has different contents. In this system, *DCC* is used to link the consumer and card-issuing bank.

2.2 Related Works

Recently, the convenience and security of wireless communication have been greatly improved. Many people enjoy online shopping with their credit cards. But due to the infrastructure of a wireless system, the transactions issued are often created via wireless. On the other hand, credit card fraud nowadays is serious, which significantly reduces online shopping attraction for some people. To enhance the security of online shopping, many m-commerce mechanisms have been proposed.

Lin et al. [23] used ID-based cryptography to establish key agreement and perform entity authentication, aiming to reduce the required computational cost, and consume memory space in mobile devices, and meet the requirements for system security, i.e., the avoidance of overspending and double spending, fairness, user anonymity and privacy.

Hwang et al. [24] introduced a tamper-resistant device (e.g., smart card) issued by a bank for authenticating payment messages sent by customers and verifying the legality of on-line payment messages. This system makes transactions possibly to be completed on-line. Virtually, the bank is not involved in the payment process.

Thus, the payment process will be more efficient and the cost per payment is reduced.

Li and Zhang [25] presented a small chip which is embedded in a credit card so as to perform the hash computation and store the previous CCT(one-time credit card transaction number). To facilitate credit card transactions, a smart card reader is needed to empower the generation of a CCT. The authors utilized a hash function to generate one-time credit card numbers as a secret which is only known to the card holder and issuer. The advantage of the system is that it effectively reduces burdens for credit card issuers, and can be easily deployed in on-line or off-line payment scenarios.

Shedid and Kouta [8]proposed a scheme to minimize the extensive computations of SET protocol by substituting the time consuming public key encryption and decryption algorithms with a symmetric key cryptography.

Chapter 3

The Proposed System

In this chapter, we will introduce the SMCS. Section 3.1 describes system parameters and functions employed in the SMCS. Section 3.2 presents the system pre-procedure before wireless communication starts. Section 3.3 lists the trading steps and their working principles.

3.1 System Parameters and Functions

3.1.1 Parameters

The system parameters utilized in the SMCS are listed below.

- *UserID*: consumer's ID.
- e, d, N: RSA encryption/decryption keys for an individual consumer.
- *Card No.*: the credit card number of the consumer.
- C_{AK}: the consumer's authentication key.
- M_{AK}: the merchant's authentication key.
- B_{AK}: the card-issuing bank's authentication key.

- PW: the password given by the consumer.
- K_{PW}: the password key derived from PW through a one-way hash function.
- *Data Connection Core* (DCC): the set of parameters that pre-link the consumer and card-issuing bank. on the mobile device side: *UserID*, e, N, K_{PW}, C_{AK} on the card-issuing bank side: *UserID*, e, d, N, *Credit-Card No.*, PW, K_{PW}, C_{AK}.
- X_a, X_{a1}, X_{a2}, X_{a3}: the consumer's private keys.
- P_{Xa}: the consumer's public key.
- X_b, X_{b1}: the merchant's private keys.
- P_{Xb}: the merchant's public key.
- CSK: the common secret key shared by the consumer and merchant.
- *Credit card's dynamic authentication code*: $\mathrm{E}_{\mathrm{AES}}(X_{a3} \oplus C_{AK}; X_{a2})$.
- *Merchant's dynamic authentication code*: $\mathrm{E}_{\mathrm{AES}}(X_{a1} \oplus M_{AK}; CSK)$.
- *Consumer's order number*: the number generated by the merchant for the consumer's online m-commerce order.
- M_{date}, M_{time}: the date and time on merchant side when it receives the consumer's m-commerce order.
- *Pre-purchase items*: the format is the consumer's order number//M_{date}//M_{time}//shopping list, where // represents concatenation.
- *Consumer's order confirmation message*: the format of this message is the consumer's order number//M_{date}//M_{time}//merchant's code, where the merchant's code is an authorization code issued by the merchant's acquiring bank.
- *Shopping association message*: the format is *consumer's order confirmation message*//business registration certificate//shopping items detail//total amount//merchant's acquiring bank's code// POS No.//$\mathrm{E}_{\mathrm{AES}}(X_{a1} \oplus$

$M_{AK}; CSK$), where business registration certificate is issued by a trustable organization or government.

- *M-commerce payment request message*: the format is $CSK//X_{a1}//\mathrm{E}_{\mathrm{AES}}(X_{a1} \oplus M_{AK}; CSK)$//*consumer's order confirmation message*//business registration certificate//total amount//merchant's acquiring bank's code//POS No.
- T_{nonce}: the timestamp of current time.
- K_{CT}: A current-time encryption key which is defined as a sequence obtained by concatenating the following current-time items, including nanosecond, second, minute, hour, month, and year, and duplicating the above sequence again when necessary to make $|K_{CT}|$ = the key length of the underlying system.
- *Trading result message*: indicating the trading success or failure. If it is trading failure, the reason is then generated and added.
- B_{date}, B_{time}: the date and time when the card-issuing bank authorizes a transaction.

3.1.2 Functions

The functions employed by the SMCS are defined below.

- Exclusive-or operator $\oplus$:

  Encryption: $c = p \oplus K$

  Decryption: $p = c \oplus K$

- Binary adder operator $+_2$:

  Encryption: $c = p +_2 K$, where plaintext p and encryption key K undergo the binary addition, ignoring the overflow bit;

  Decryption: $p = c -_2 K = \begin{cases} c - K, if c \geq K \\ c + \bar{K} + 1, if c < K \end{cases}$

- RSA encryption/decryption function:

  Encryption: RSA_En$(x, e) = x^e \bmod N$, where x is a random key.

  Decryption: RSA_En$(y, d) = y^d \bmod N$, where $y =$ RSA_En(x, e)

- En1$(a, b; x) = (x \oplus a) +_2 b$, where x is the a key to be protected, and a and b are encryption keys.

  InvEn1$(a, b; y) = (y -_2 b) \oplus a$ is the inverse function of En1(), where $y=$ En1$(a, b; x)$.

- En2$(a, b; str) = (s_1 \oplus a) +_2 b//(s_2 \oplus a) +_2 b//(s_3 \oplus a) +_2 b//.../(s_n \oplus a) +_2 b$, in which a and b are encryption keys and $str = s_1//s_2//s_3//\ldots//s_n$.

  InvEn2$(a, b; Cstr) = (cs_1 -_2 b) \oplus a//(cs_2 -_2 b) \oplus a//(cs_3 -_2 b) \oplus a//\ldots//(cs_n -_2 b) \oplus a$, in which a and b are decryption keys and Cstr= En2$(a, b; str) = cs_1//cs_2//cs_3//\ldots//cs_n$.

  For example: in this system, the credit card's dynamic authentication code $\mathrm{E_{AES}}(X_{a3} \oplus C_{AK}; X_{a2})$ as shown in Figure 3 is generated by invoking AES and inputting the consumer' s random dynamic key X_{a2} and the result of exclusive-oring X_{a3} and the consumer's authentication key C_{AK}.

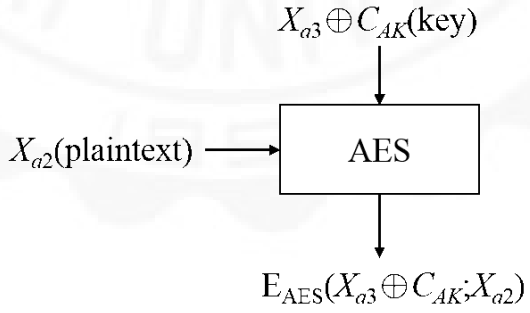

FIGURE 3: The process of generating the dynamic authentication code for a credit card.

- *OP_code*: In the SMCS, different messages are generated for different purposes. Each message has its own unique operation code (*OP_code* for short) to indicate the designate function of the message. It can reduce the authentication time and complexity. Table1 lists definitions of the employed *OP_code*.

- *Status*: each subsystem installed in the consumer, merchant, and card-issuing bank has its own internal parameter (i.e., *status*), which is used to indicate the state that the SMCS will achieve at the next moment. When *status* is used in conjunction with *OP_code*, they can effectively improve system performance on certification, and protect the underlying system against replay attacks. Table 1 also lists the statuses and their descriptions.

TABLE 1: Definitions of employed *OP_codes* and the corresponding *statuses*.

OP_code	*Status*	Description
1	1	M-commerce requirements
2	2	M-commerce reply
3	3	M-commerce order
4	4	M-commerce order confirmation
5	5	M-commerce payment request
6	6	M-commerce payment reply
7	7	Card-issuing bank payment
8	8	Electronic invoice
9	9	Completion of the transaction

- HMAC(K): a specific integrity function employing a cryptographic hash function and a secret key K to produce a hash-based message authentication code (HMAC for short) which ensures accuracy, integrity, and non-repudiation of the corresponding message [26] [27].

3.2 Pre-procedure

Each of card-issuing bank, merchant and consumer has its own pre-procedure.

3.2.1 The card-issuing bank

The card-issuing bank initially submits an identity-authentication-key application to the CA. CA generates an authentication key B_{AK} and sends the key to the card-issuing bank. After that, the card-issuing bank and CA establish a communication channel by using B_{AK} and communicate with each other through the channel.

3.2.2 The merchant

The merchant who has a qualified Business registration certificate issued by government (or a trustable organization) sends a message to CA to apply for an identity authentication key. After inspecting the application documents and confirming that the merchant is a legitimate company, CA creates an authentication key M_{AK} and deliveries the key to the merchant. After that, CA periodically contacts the merchant to make sure the legitimation of the merchant.

3.2.3 The consumer

The consumer's pre-procedure has three steps.

(1) The consumer applying for the *DCC* from the card-issuing bank.

The consumer applies for a *DCC* from the card-issuing bank with over-the-counter service. But, the consumer needs to provide his/her personal information, including user name, personal ID, birthday, residence address, email address, photocopy of the front and back of him/her identity card (in Taiwan), proof of financial statement and his/her own password (PW).

(2) Generating the *DCC*

If the card-issuing bank confirms the identity of the credit card owner, then depending on the consumer's credit card number and password, the bank creates the consumer's *DCCs*, issues the consumer's *DCC*, i.e., $(UserID, e, N, K_{PW}, C_{AK})$,

to the consumer's mobile device, and keeps the *DCC* for the consumer, i.e., $(UserID, e, d, N, CardNo., PW, K_{PW}, C_{AK})$, in its local database.

(3) The m-commerce APP program is downloaded to the mobile device.

3.3 Trading Steps and Working Principles

Figure 4 illustrates an overview of the communication steps of the SMCS. Steps $1.1 \sim 1.4$ comprise the m-commerce order confirmation stage. Steps 2.1 and 2.2 are the m-commerce payment stage. The card-issuing bank authorization stage includes Steps 3.1 and 3.2. Step 4 itself is the electronic invoice delivery stage.

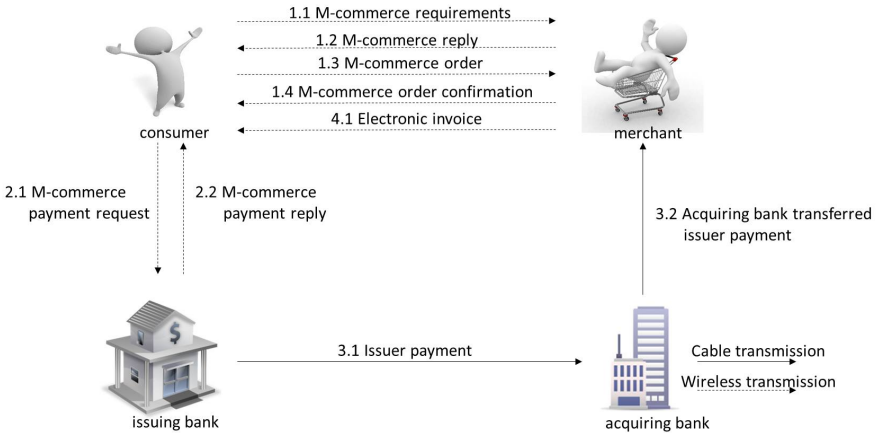

FIGURE 4: The communication steps of the SMCS.

Pre-procedure for m-commerce:

(1) When the mobile device is in its standby mode, it activates the m-commerce APP installed in it.

(2) Visiting and browsing the merchant's web page under the guidance of the APP.

3.3.1 The m-commerce order confirmation stage

Step 1.1: The consumer

After browsing the merchant's web page, the consumer moves all his/her favorite products into the shopping cart, and sends a message, denoted by Message 1 carrying the shopping list, to the merchant. The format of this message is as follows. $OP_code|P_{Xa}|$shopping list

in which $OP\_code = 1$, and P_{Xa} is the consumer's public key. Also, the consumer's *status* is set to 2.

Step 1.2: The merchant

After receiving this message, the merchant uses its own private key X_b and the consumer's public key P_{Xa} to calculate the common secret key CSK, where $CSK = P_{Xa}{}^{Xb} mod\ p$ and sends a m-commerce reply, denoted by Message 2, to the consumer. The format of Message 2 is as follows.

$OP_code|P_{Xb}|CSK \oplus X_{b1}|$En2$(CSK, X_{b1};$ *pre-purchase items*$)|$HMAC$(CSK +_2 X_{b1})$

in which $OP\_code = 2$. The merchant then sets *status* to 3 as the *status* of the consumer's m-commerce order.

Step 1.3: The consumer

After receiving Message 2, the consumer checks to see whether $OP\_code \stackrel{?}{=} status$.If not, it discards this message and waits for a valid one. Otherwise, the consumer knows that the message is a m-commerce reply and then computes CSK where $CSK = P_{Xb}{}^{Xa} mod\ p$. The consumer further decrypts $CSK \oplus X_{b1}$ by using CSK to obtain X_{b1} and verifies the message by checking to see whether HMAC$(CSK +_2 X_{b1})_c \stackrel{?}{=}$ HMAC$(CSK +_2 Xb1)_r$where the subscript $_c$ means that

the HMAC() is derived from the consumer's internal parameters, and the subscript r represents that the HMAC() is retrieved from Message 2. If they are not equal, the consumer discards this message and waits for a valid one. Otherwise, the consumer uses CSK and X_{b1} to decrypt En2(CSK, X_{b1}; *pre-purchase items*) to recover the *pre-purchase items* which include the consumer's order number, M_{date}, M_{time} and shopping list. If the consumer does not confirm the list, the process goes back to Step 1.1. Otherwise, the consumer sends Message 3 to the merchant. The format of this message is as follows.

OP_code|consumer's order number|En1($CSK, X_{b1}; X_{a1}$)|En2(CSK, X_{a1}; consumer name//delivery address//shopping items detail//consumer phone number)| HMAC($X_{a1} +_2 CSK$)

in which *OP_code* = 3. The consumer's *status* is then set to 4.

Step 1.4: The merchant

When receiving Message 3, the merchant retrieves the *Consumer's order number* from this message and the consumer's *status*, and tests whether *OP_code* $\stackrel{?}{=}$ *status*.If not, the merchant discards the message and waits for a valid one. Otherwise, meaning this is a m-commerce order, the merchant decrypts En1($CSK, X_{b1}; X_{a1}$) by using CSK and X_{b1} to obtain X_{a1}, and certifies the message by testing whether HMAC($X_{a1} +_2 CSK)_c \stackrel{?}{=}$ HMAC($X_{a1} +_2 CSK)_r$ where the subscript $_c$ means HMAC($X_{a1} +_2 CSK$) is calculated by using the merchant's internal parameters, and the subscript r represents that HMAC($X_{a1} +_2 CSK$) is retrieved from Message 3. If they are not equal, the merchant discards this message and waits for a valid one. Otherwise, the merchant uses CSK and X_{a1} to decrypt the message to obtain the consumer name, delivery address, shopping items detail and consumer phone number. It then sends the m-commerce order confirmation message (Message 4) to the consumer. The format of Message 4 is as follows.

OP_code|En2(X_{a1}, X_{b1}; *shopping association message*)|HMAC(($CSK \oplus Xb1) +_2$ X_{a1})

in which $OP\_code = 4$. The merchant's *status* is set to 7.

3.3.2 The m-commerce payment stage

Step 2.1: The consumer

When receiving Message 4, the consumer tests whether $OP\_code \stackrel{?}{=} status$. If not, the consumer discards the message and waits for a valid one. Otherwise, showing that the message is the m-commerce order confirmation, the consumer certifies the message by testing whether $\text{HMAC}((CSK \oplus X_{b1}) +_2 X_{a1})_c \stackrel{?}{=} \text{HMAC}((CSK \oplus X_{b1}) +_2 X_{a1})_r$ where the subscripts $_c$ and $_r$ are respectively the same as those mentioned above. If they are not equal, the consumer discards this message and waits for a valid one. Otherwise, he/she uses X_{a1} and X_{b1} to dcrypt $\text{En2}(X_{a1}, X_{b1}$;*shopping association message*) to obtain *shopping association message* which contains the *consumer's order confirmation message*, business registration certificate, shopping item detail, total amount, merchant's acquiring bank's code, POS No., and $\text{E}_{\text{AES}}(X_{a1} \oplus M_{AK}; CSK)$. If the consumer confirms the information and is ready to purchase, the m-commerce APP will ask the consumer to input his/her own password (PW), accordingly compute the corresponding password key $K_{pw,c}$ based on the established algorithms beforehand and then compare the key with K_{pw}, i.e., the consumer's password key stored in the DCC. If they are not equal, the APP asks the user to input the password again. If the user cannot pass the authentication for three times, the system will shut off the m-commerce APP. If login is successful, the consumer sends the m-commerce payment request message, denoted by Message 5, to the card-issuing bank. The format of this message is as follows.

$$OP\_code|T_{nonce}|UserID|\text{RSA\_En}(X_{a2}, e)|\text{En1}(X_{a2}, X_{a2} \oplus K_{pw}; X_{a3})|$$
$$\text{En2}(X_{a2}, X_{a3} \oplus K_{CT}; \textit{m-commerce payment request message})|X_{a1} \oplus \text{E}_{\text{AES}}(X_{a3} \oplus C_{AK}, X_{a2})|\text{HMAC}((K_{CT} \oplus X_{a2}) +_2 X_{a3})$$

in which $OP\_code = 5$. The consumer's *status* is set to 6.

Step 2.2: The card-issuing bank

Upon receiving Message 5, the card-issuing bank retrieves the consumer's *status* (the value is 5) stored in the card-issuing bank's system based on the $UserID$, and tests whether $OP\_code \stackrel{?}{=} status$ (the first authentication). If not, the card-issuing bank discards the message and waits for a valid one. Otherwise, the card-issuing bank verifies $T_{received} - T_{nonce} < \Delta T$(the second authentication). If not, it discards the message. If yes, it derives K_{CT} from T_{nonce}. Furthermore, by $UserID$, the card-issuing bank retrieves the consumer's RSA encryption/decryption key (e, d, N) from the consumer's DCC, decrypts RSA_En(X_{a2}, e) to obtain X_{a2}, decrypts En1$(X_{a2}, X_{a2} \oplus K_{pw}; X_{a3})$ by using X_{a2} and K_{pw} to recover X_{a3}, and at last decrypts En2$(X_{a2}, X_{a3} \oplus K_{CT}$; *m-commerce payment request message*) by utilizing X_{a2}, X_{a3} and K_{CT} to obtain the *m-commerce payment request* message.

The *m-commerce payment request* message which conveys $CSK, X_{a1}, \mathrm{E}_{\mathrm{AES}}(X_{a1} \oplus M_{AK}, CSK)$, *consumer's order confirmation message* (from which to retrieve consumer's order number), business registration certificate, total amount, merchant's acquiring bank's code and POS No. It performs the third authentication by testing whether $\mathrm{HMAC}((K_{CT} \oplus X_{a2}) +_2 X_{a3})_c \stackrel{?}{=} \mathrm{HMAC}((K_{CT} \oplus X_{a2}) +_2 X_{a3})_r$. If they are not equal, the bank discards this message and waits for a valid one. Otherwise, according to the business registration certificate, the card-issuing bank retrieves M_{AK} from its database. If M_{AK} does not exist in its database, the bank will ask CA for M_{AK}, and store it in the card-issuing bank's database. After obtaining M_{AK}, the card-issuing bank performs the fourth authentication by checking to see whether $\mathrm{E}_{\mathrm{AES}}(X_{a1} \oplus M_{AK}, CSK)_c \stackrel{?}{=} \mathrm{E}_{\mathrm{AES}}(X_{a1} \oplus M_{AK}, CSK)_r$. If they are not equal,it discards this message and waits for a valid one. Otherwise, the card-issuing bank uses X_{a1} to decrypt $X_{a1} \oplus \mathrm{E}_{\mathrm{AES}}(X_{a3} \oplus C_{AK}, X_{a2})$ carried in Message 5 to obtain the credit card's dynamic authentication code $\mathrm{E}_{\mathrm{AES}}(X_{a3} \oplus C_{AK}, X_{a2})_r$, retrieves C_{AK} from the consumer's DCC to generate $\mathrm{E}_{\mathrm{AES}}(X_{a3} \oplus C_{AK}, X_{a2})_c$, and then performs the fifth authentication by testing whether $\mathrm{E}_{\mathrm{AES}}(X_{a3} \oplus C_{AK}, X_{a2})_c \stackrel{?}{=} \mathrm{E}_{\mathrm{AES}}(X_{a3} \oplus C_{AK}, X_{a2})_r$. If they are not equal, the bank discards the message and waits for a valid one. Otherwise, the bank

retrieves the consumer's credit card number from the consumer's DCC. After checking the consumer's credit, the card-issuing bank determines whether to authorize the transaction. Also, no matter whether the transaction is successfully authorized or not, the card-issuing bank will inform the consumer of the trading results by sending the m-commerce payment reply (Message 6) to the consumer. The format of Message 6 is as follows.

$$OP\_code|\text{En2}(CSK, X_{a1};\ \textit{consumer's order confirmation message//trading results message}//B_{date}//B_{time})|\text{HMAC}(X_{a3} \oplus X_{a2})$$

in which $OP\_code = 6$. The card-issuing bank further sends Message 7 to the acquiring bank. The acquiring bank will transfer this message to inform the merchant of the trading results. If the transaction fails, the card-issuing bank provides reasons for the failure in Message 7. The format of this message will be described later.

When receiving Message 6, the consumer tests whether $OP\_code \stackrel{?}{=} status$. If not, the consumer discards the message and waits for a valid one. Otherwise, meaning the message is the m-commerce payment reply, the consumer certifies the message by testing whether $\text{HMAC}(X_{a3} \oplus X_{a2})_c \stackrel{?}{=} \text{HMAC}(X_{a3} \oplus X_{a2})_r$. If they are not equal, the consumer discards this message and waits for a valid one. Otherwise, the consumer employs CSK and X_{a1} to decrypt the message to obtain the *consumer's order confirmation message*, *trading results message*, B_{date} and B_{time}. If the transaction fails, the reason for the failure is shown and this transaction is terminated. Otherwise, the consumer sets $status = 8$, and waits for the merchant to send the electronic invoice.

3.3.3 The Card-issuing bank authorization stage (via a wired communication channel)

Step 3.1: The card-issuing bank

The card-issuing bank payment message (Message 7) is sent by the card-issuing bank to the acquiring bank via a wired credit-card communication channel. Therefore, the message and content are encrypted by using the channel' s encryption method. The format of this message is as follows:

OP_code|*consumer's order confirmation message*|POS No.|*trading results message*|card number|authorization code|total amount|B_{date}|B_{time}

in which $OP\_code = 7$.

Step 3.2: The acquiring bank

When the acquiring bank receives Message 7, it identifies the merchant of this transaction based on the merchant's code and POS No., and then transfers the message to the merchant to inform him/her of the trading results (Message 8). The format of this message is as follows.

OP_code|*consumer's order confirmation message*|authorization code|*trading results message*|B_{date}|B_{time}|POS No.

in which $OP\_code = 7$. When receiving the message, this merchant retrieves this transaction's *status* according to the *consumer's order number* (carried in *consumer's order confirmation message*), and tests whether $OP\_code \stackrel{?}{=} status$. If not, the merchant discards the message and waits for a valid one. Otherwise, the merchant knows that the message is the card-issuing bank payment, which includes the trading results. If the transaction fails, the merchant terminates this transaction. Otherwise, products and the electronic invoice are shipped to the consumer by executing Step 4.

3.3.4 The electronic invoice delivery stage

Step 4 The merchant

The merchant sends the electronic invoice (Message 9) to the consumer. The format of this message is as follows:

$$OP\_code|electronic\ invoice|\text{HMAC}(CSK +_2 X_{a1} \oplus X_{b1})$$

in which $OP\_code = 8$. The merchant also sets *status* of the consumer's order number to 9. After receiving the message, the consumer tests whether $OP\_code \stackrel{?}{=} status$. If not, the consumer discards the message and waits for a valid one. Otherwise, the consumer tests whether $\text{HMAC}(CSK +_2 X_{a1} \oplus X_{b1})_c \stackrel{?}{=} \text{HMAC}(CSK +_2 X_{a1} \oplus X_{b1})_r$. If not, the consumer discards the message and waits for a valid one. Otherwise the consumer saves the electronic invoice, and sets $status = 9$ to complete this transaction.

Chapter 4

Performance and Security Analysis

In the following, we will analyze and discuss features of the SMCS on security and performance.

4.1 Performance Evaluation

The SMCS was simulated in a card-issuing-bank-consumer–merchant environment. The program is developed by using Java. The hardware specifications of the test-bed are listed in Table 2.

TABLE 2: The specifications of the test-bed utilized to simulate the consumer device, merchant and card-issuing bank.

Component	Card-issuing bank	Consumer	Merchant
CPU	Intel i7-950 3.07GHz	Intel i5 2.67GHz	Intel i7-950 3.07GHz
RAM	12 GB	2 GB	12 GB
Platform	Windows 7, 64-bit	Windows 7, 32-bit	Windows 7, 64-bit

The simulation includes the internal-key generation, communication-key generation and message generation given different key lengths, including 512, 768 and 1024 bits, and different keys, including consumer's private keys (X_a, X_{a1}, X_{a2} and X_{a3}), merchant's private keys (X_b and X_{b1}), Diffie–Hellman PKDS public keys (P_{Xa} and P_{Xb}), common secret key (CSK) and current-time encryption key (K_{CT}).

Table 3 lists the simulation results, in which the times required to generate private keys and current-time encryption key were small, but the times consumed to generate Diffie–Hellman PKDS public keys and common secret key are relatively long, indicating that they are the dominated factors for the SMCS performance.

TABLE 3: The timings required to generate the internal keys and communication keys on the card-issuing bank, consumer and merchant ends in the SMCS.

Key	Key generation time (μs)					
	Consumer			Merchant and Card-issuing bank		
	Size (bits)					
	512	768	1024	512	768	1024
X_a	0.58	0.71	0.95	-	-	-
X_{a1}	0.52	0.71	0.91	-	-	-
X_{a2}	0.55	0.70	0.91	-	-	-
X_{a3}	0.52	0.75	0.90	-	-	-
X_b	-	-	-	0.50	0.67	0.82
X_{b1}	-	-	-	0.47	0.67	0.81
P_{Xa}	762.32	2331.59	4991.25	-	-	-
P_{Xb}	-	-	-	175.13	495.98	1002.35
CSK	2563.11	7935.23	17691.32	552.71	1652.26	3531.44
K_{CT}	3.69	3.87	4.00	-	-	-

Remark : (-: does not exist)

Table 4 illustrates the timings required to execute different operators and functions utilized in this study, including exclusive-or, binary adder, binary subtract, En1(), En2(), HMAC(), RSA() and $E_{AES}()$. We can see that HMAC(), RSA() and $E_{AES}()$ consume the longest times, indicating that we have to reduce their usage to improve the system performance.

TABLE 4: The timings required to calculate different operators and functions utilized in the SMCS.

Function or operator	Key generation time (μs)					
	Consumer			Merchant and Card-issuing bank		
	Size(bits)					
	512	768	1024	512	768	1024
$\oplus$	0.21	0.27	0.37	0.14	0.20	0.23
$+_2$	0.71	1.05	1.44	0.49	0.71	0.93
$-_2$	1.95	2.83	3.65	0.84	1.22	1.59
En1()	0.87	1.30	1.69	0.64	0.87	1.13
En2()	2.70	3.81	5.00	1.74	2.50	3.28
HMAC()	112.31	125.09	139.6	39.81	41.36	47.76
RSA()	2512.36	7801.25	17385.91	-	-	-
$E_{AES}()$	79.31	81.11	87.94	23.90	24.81	27.46

The numbers of operations required to produce data carried in a message are shown in Table 5. This table also lists the computation times for the generation of a message given 512, 768, and 1024 bits as the key lengths. Since Message 1 conveys OP_code, P_{Xa}, and shopping list, the time consumed is almost equal to the time required to produce P_{Xa}. Other messages have the similar phenomenon.

TABLE 5: The operations that constitute a message in the SMCS encryption/ decryption processes and computation times for the generation of messages.

Message	Number of operations	Time consumed (ms)		
		Size (bits)		
		512	768	1024
Message 1	R+D	0.76	2.33	4.99
Message 2	2R+D+C+X+En2+H+B	0.77	2.19	4.58
Message 3	En1+C+R+En2+H+B	2.68	8.06	17.83
Message 4	En2+E_{AES}+H+2X+B	0.06	0.06	0.08
Message 5	RSA+En1+2R+En2+T+5X+E_{AES}+H+B	2.71	8.02	17.62
Message 6	En2+X+H	0.04	0.04	0.05
Message 9	H+B+X	0.04	0.04	0.05

R: random number generation

D: Diffie-Hellman public key generation

C: common secret key (CSK) generation

X: exclusive-or operation

B: binary addition

S: binary subtraction

T: K_{CT} generation

H: HMAC() generation

RSA: RSA() encryption

En1: En1() encryption

En2: En2() encryption

E_{AES}: E_{AES}() encryption

Basically, Message 5 is strongly related to the security of payment information. It needs to be securely protected. So this message is generated spending the longest time among all messages. Message 6 generated by the card-issuing bank and Message 9 produced by the merchant only use binary adder and exclusive-or, and generate HMAC(). So their generation times are short. Message 7 and Message 8 are transmitted through the credit card system's wired communication channel so we did not analyze them in this study. That is why they do not appear in Table 5.

4.2 Security Evaluation

The SMCS has the following nine features which make a m-commerce system more secure, efficient and convenient.

(1) When a wireless communication environment is available, users, even outdoor, can use the SMCS to conduct m-commerce, meaning it is a convenient shopping environment.

(2) The merchant does not know the consumer's credit card number and its related information, and the card-issuing bank does not know the consumer's order list. Thus, the shopping behavior has been highly secured.

(3) A consumer links himself/herself to the card-issuing bank with the consumer's *DCC*, thus forming a closed-security architecture to protect the m-commerce environment.

(4) Through CA certification, the SMCS ensures the legality of a merchant (rather than a faked one) to protect the interests and shopping behaviors of consumers.

(5) As an open architecture between a consumer and a merchant, the SMCS uses Diffie-Hellman PKDS to establish a safe and convenient security mechanism for the two entities to communicate with each other.

(6) The encryption function $\mathrm{En2}(a, b; str)$ is used to encrypt plaintext stream str. Such a two-dimensional stream cipher scheme can effectively improve the security of a stream cipher technique.

(7) C_{AK} and M_{AK}, which serve as mother keys, are derived from the credit card's and merchant's dynamic authentication codes, thus greatly enhancing the safety of m-commerce since the two codes are highly secured.

(8) In the SMCS, all delivered messages, except the one sent in Step 1, are protected by $\mathrm{HMAC}(K_d)$, in which the key K_d is derived from the dynamic linking keys created by the two sides of a wireless connection for securing messages transmitted between them so as to ensure the communication privacy, integrity, non-repudiation and mutual authentication of the delivered messages.

(9) The security of Message 5 determines whether the payment request is securely transmitted or not. So this message needs to be protected by PW. In fact, it has five authentication mechanisms, including authenticating OP_code, checking T_{nonce}, verifying HMAC(), certifying merchant and certifying consumer. Therefore, we can ensure that Message 5 is a legitimate m-commerce payment request. This also ensures safety of the SMCS.

In the SMCS, the security mechanism employed at the m-commerce order confirmation stage is the Diffie-Hellman PKDS which is very difficult to be cracked [28]. Hence, the generated communication keys including CSK, X_{a1} and X_{b1} are secure. However, the information of the pre-purchase items (rather than the items themselves), including consumer name//delivery address//shopping item data// consumer phone number and the shopping association message, is protected by

En2(). The security levels of En1() and En2() will be analyzed in Theorem 1 under the assumption that the keys used in the SMCS are all n-bits in length where n = 128, 256, 512, 1024 or 2048.

Theorem 1:

Let $y = \text{En1}(a, b; x)$ be the ciphertext key generated by encrypting plaintext key x with encryption keys a and b. If (plaintext, ciphertext) pair, denoted by (x, y), is known by hackers, the probability with which hackers can obtain the correct encryption pair, i.e., (a, b), is $\frac{1}{2^n}$. Furthermore, if only the ciphertext key y is known by hackers, the plaintext key x is also practically secure.

<Pf>

First, we rewrite y = $\text{En1}(a, b; x) = (a \oplus x) +_2 b$ as

$$y -_2 b = a \oplus x \tag{1}$$

in which two linearly independent unknown keys a and b are employed to encrypt known keys x and y with two operators $\oplus$ and $+_2$. For each unknown key, there are 2^n possible values, implying that for each known pair (x, y), there are 2^n possible pairs of (a, b)s that satisfy Eq.(1) ، since for each a (b), there exist 2^n bs (as) that make Eq.(1) true. Hence, the probability with which to crack $\text{En1}(a, b; x)$ when (plaintext, ciphertext) pair is known and then obtain the correct (a, b) pair is $\frac{1}{2^n}$.

Furthermore, if only the ciphertext key y is known by hackers, Eq.(1) can be rewritten as

$$x = (y -_2 b) \oplus a \tag{2}$$

In which x is derived from three keys, i.e., a, b and y, by using $-_2$ and $\oplus$ operators. For each known ciphertext key y, there are $(2^n)^2 = 2^{2n}$ possible a, b and x key-combinations that meet Eq.(2) since when one of the three parameters, e.g., x, is known, there are $2^{2n}(b, y)$ pairs that will satisfy Eq.(2). Hence, the probability with which to obtain the correct value of (a, b, x) triple from known y is $\frac{1}{2^{2n}}$. In other words, when hackers know ciphertext y, the probability with which they can obtain correct plaintext x by breaking Eq.(2) is $\frac{1}{2^{2n}}$which is less than $\frac{1}{2^n}$, the probability of a blind guess on x on one trial. Hence, plaintext key x practically secure, even though ciphertext key y is known by hackers.

In En2$(a, b; str)$, like that in En1$(a, b; x)$, the encryption keys are also a and b. But the plaintext $str = p_1//p_2//\ldots//p_m$ is a stream conjunction keys, meaning a sub-stream P_i conjunctively exists with other sub-streams P_j, rather than an independent one. So the ciphertext is also a stream conjunction keys, i.e., $Cstr =$ En2$(a, b; str) = c_1//c_2//\ldots//c_m$ where

$$c_j = (a \oplus p_j) +_2 b, 1 \leq j \leq m \tag{3}$$

In the SMCS, each time when En2$(a, b; str)$ is invoked, the plaintext str is protected by randomly generated encryption key pair (a, b). Hackers only know the ciphertext stream $Cstr$ and do not know (a, b). Then, by Theorem 1 and Eq. (3), the plaintext stream $p_1//p_2//\ldots//p_m$ is well protected.

When the consumer completes the e-commerce order confirmation stage, he/ she may start the transmission of the m-commerce payment request, i.e., Message 5, or delay/cancel this m-commerce. In Message 5, the random key X_{a2} is protected by using RSA_En(X_{a2}, e) which is sufficiently secure since the RSA encryption key pair (e, N) is a part of the consumer's DCC. Hence, hackers cannot acquire it. Moreover, the random key X_{a3} is protected by En1$(X_{a2}, X_{a2} \oplus K_{PW}; X_{a3})$ and is also sufficiently secure since by Theorem 1, hackers do not know X_{a2} and $X_{a2} \oplus K_{PW}$. Thus, the random keys X_{a2} and X_{a3} that link the consumer and the card-issuing bank are securely protected when they are carried in Message 5.

Also, in Message 5, the credit card's dynamic authentication code $\mathrm{E}_{\mathrm{AES}}(X_{a3} \oplus C_{AK}; X_{a2})$ is protected by X_{a1}. However, even if $\mathrm{E}_{\mathrm{AES}}(X_{a3} \oplus C_{AK}; X_{a2})$ is known by hackers, the consumer's authentication key C_{AK} is still secure. Any forged Message 5 will not pass the authentication performed by the card-issuing bank. Theorem 2 will show this.

Theorem 2:

In the SMCS, the security mechanisms of the m-commerce payment request, i.e. Message 5, can effectively defend three common attacks, i.e., forgery attack, replay attack and eavesdropping attack.

<Pf>

First, anyone, including a hacker, who does not have DCC, will not own (e, N) and K_{PW}, implying that the encryption codes, RSA_En(X_a2, e) and En1$(X_{a2}, X_{a2} \oplus K_{PW}; X_{a3}$, generated by hackers cannot be correctly decoded by the card-issuing bank. That is, $X_{a2,h} \neq X_{a2,b}$ and $X_{a3,h} \neq X_{a3,b}$ where subscript h (b) represents that the random keys X_{a2} and X_{a3} are generated by hackers (obtained by card-issuing bank through decoding). Hence, the two forged authentication codes, i.e., HMAC$((K_{CT} \oplus X_{a2}) +_2 X_{a3})$ and $\mathrm{E}_{\mathrm{AES}}(X_{a3} \oplus C_{AK}, X_{a2})$, carried in Message 5 cannot pass the authentication performed by the card-issuing bank. That is, the m-commerce payment request can effectively defend the forgery attack.

Second, with the T_{nonce} and HMAC$((K_{CT} \oplus X_{a2}) +_2 X_{a3})$ carried in Message 5, the SMCS can effectively defend a replay attack [21] [29].

At last, by $UserID$, hackers can collect a particular consumer's communication messages, especially Message 5, within a period of time. However, for each communication, $T_{nonce}, K_{CT}, X_{a2}$ and X_{a3} are changed randomly so that the parent key of AES, i.e., $X_{a3} \oplus C_{AK}$ and plaintext X_{a2} (see Figure 3) vary also on different m-commerce transactions (or just transactions). The result is that the ciphertext of AES, i.e., $\mathrm{E}_{\mathrm{AES}}(X_{a3} \oplus C_{AK}; X_{a2})$, varies randomly on different transactions. Thus, the statistical analysis on the collection of $X_{a2}, X_{a3} \oplus C_{AK}$ and

$E_{AES}(X_{a3} \oplus C_{AK}; X_{a2})$ is useless since, for each transaction, $X_{a2}, X_{a3} \oplus C_{AK}$ and $E_{AES}(X_{a3} \oplus C_{AK}; X_{a2})$ are individually unique and independent. It is hard for hackers to decrypt AES so as to obtain the right plaintext X_{a2} from known ciphertext $E_{AES}(X_{a3} \oplus C_{AK}; X_{a2})$ if the parent key $X_{a3} \oplus C_{AK}$ is unknown, indicating that the m-commerce payment request can effectively defend the eavesdropping attack.

Although there are many other possible attacks, such as DOS, main-in-the-middle, wormhole attacks, etc. that threaten the security of wireless network communication. However, for the one way propagation property, from consumer to its card-issuing bank and the security mechanisms of the m-commerce payment request, Message 5 can effectively defend against them, and hence the consumer' s interests are well protected.

In this study, since the payment mechanism completely rules out the involvement of trading merchant, and credit card number does not appear in the wirelessly transmitted messages, credit card fraud can be effectively avoided. This is another advantage of the SMCS.

Next, we will discuss two security issues of m-commerce transactions. (1) Is it possible for a consumer to send a fake m-commerce payment request message to deceive card-issuing bank's money, then deny the transaction afterward and pretend that he/she is also a victim? If the answer is yes, it will be a serious security problem. (2) If a consumer's mobile phone is lost, one would like to know whether anyone who finds the phone can counterfeit the legitimate consumer to wirelessly shop via this mobile phone or not. If the answer is positive, it will also be another serious security problem. Theorem 3 will show that the SMCS does not have such problems.

Theorem 3:

Only legitimate mobile phone owner's wireless m-commerce messages can pass all authentications, i.e., an invalid m-commerce payment request message cannot pass

the authentication performed by all security mechanisms.

<Pf>

By Theorem 2, a fake m-commerce payment request message cannot pass certification if the hacker does not know the consumer's DCC. Therefore, only the consumer himself/herself can create a fake m-commerce payment request message (i.e, Message 5) in an attempt to pass the certification. However, in the SMCS, consumer's DCC is encrypted and stored in his/her mobile phone, only the designate phone APP can interpret the information correctly. In other words, a consumer does not know his/her own DCC. So it is impossible for him/her to produce a fake m-commerce payment request message that can pass all authentications. Secondly, when consumers like to transact with an invalid or unqualified merchant, since the merchant cannot pass the CA certificate, so the merchant does not have a legitimate merchant's authorization key (M_{AK}) to produce the correct Merchant's dynamic authentication code $\mathrm{E}_{\mathrm{AES}}(X_{a1} \oplus M_{AK}; CSK)$ which can be successfully certified by the card-issuing bank. If an m-commerce payment request message can pass the five authentication mechanisms of the SMCS, the consumer must be a legitimate one who transacts with a qualified merchant through the mobile phone' s APP. That is, only a legitimate consumer who transacts with a qualified merchant can generate a legitimate m-commerce payment request message that can pass the mentioned five security authentication mechanisms provided, meaning the safety of the SMCS can be actually ensured.

If one, e.g., u, uses other consumer's mobile phone to purchase something through the mentioned APP, even though u can pass Step 1, i.e., the m-commerce order confirmation stage, when wishing to generate a m-commerce payment request message, u must enter the correct PW which is not stored in the mobile phone. However, without PW, u cannot start the APP to generate a m-commerce payment request message. If u sends a fake one through u's mobile phone, the message also cannot pass the authentication because u is unable to correctly decode the DCC to correctly recover (e, N), K_{PW} and C_{AK}. Therefore, the fake m-commerce payment request message cannot pass the authentication, indicating

that even though the consumer loses his/her mobile phone, the SMCS can still prevent the mobile phone finder from illegally purchasing something through the mobile phone.

Table 6 lists the security comparison among the SMCS, SSL and SET. We can see that the SMCS is more secure than the other two.

TABLE 6: The security comparison among the SMCS, SSL and SET.

	SSL	SET	SMCS
Confidentiality	√	√	√
Integrity	√	√	√
Non-repudiation	-	√	√
Peer-to-peer delivery safety	√	√	√
Certified merchant's identity	-	√	√
Sent m-commerce payment request to card-issuing bank through by who	Merchant	Merchant	Direct to the card-issuing bank
Efficacy	High	Low	Middle
Security	Low	Middle	High
Convenience	Middle	Low	High

Remark : (- : Without having this ability)

4.3 compare with other system

Table 7 lists the comparison of the SMCS with other systems..A digital wallet needs its owner to pre-deposit some amount of money into the consumer's wallet before he/she can shop something with the wallet. The micropayment limits consumers to buy low-price items, so it is not suitable for purchasing expensive ones. In addition, this is inconvenient for consumers since when shopping they need to carry a smart card. Currently many enjoy-first-pay-later shopping policies have been proposed, and the allowable amount of payment is higher. Also, it is more risk if the consumer sends an m-commerce payment request to third party through merchant rather than consumer sending the payment request directly to third party, even though this will increase the burden of mobile devices. In our system, we use *DCC* to link consumer and card-issuing bank. Consequently, the two sides do not have to spend time and efforts to establish a secure channel.

TABLE 7: comparison of the SMCS with other systems.

	SMCS	MEP [23]	Hwang et al. [24]	Li and Zhang [25]	MSET [8]
Payment methods	Credit card	e-wallet	Micro-payment	Credit card	Credit card
Payer	card-issuing bank	Third party payment	Third party payment	card-issuing bank	card-issuing bank
Requiring a smart card	no	yes	yes	yes	no
Who sends an m-commerce payment request to Third party	consumer	consumer	merchant	merchant	merchant
Purchase physical items	yes	no	yes	yes	yes

In this table, we compare the efficiencies of the SMCS, SET and MSET. All of them can be invoked directly with a credit card, and SET mechanism is now in use. From table 8, we can see that SMCS is faster than the SET. Because the mechanism of SET's parties use asymmetric key encryption to establish a secure channel. Both sides of a communication connection are very safe. But the time consume is also long. In the SMCS, consumers need to transmit encrypted messages to the merchant. The merchant reuses the encrypted messages to request card-issuing bank for payment. In order to prevent the merchant from knowing the consumer's credit card information, and card-issuing bank from knowing the consumer's shopping information, SET uses three asymmetric encryption and two symmetric encryption methods to protect consumer's data. MSET requests all its members to go to its stores to register their public/private key pair with the stores before the consumers are allowed to shop something. Before shopping, a consumer utilizes the public key to encrypt a session key, and the other side of the connection employs the private key to decrypt the session key. The session key is then used to encrypt messages exchanged between two sides of the connection. Also, In the MSET, Consumer's credit card information needs to be transmitted to the merchant. We consider that the MSET is not very efficient and security level

TABLE 8: The numbers of operations required and times consumed by the SMCS, SET and MSET.

Operation	Time consumed on 512 bits for one operation(μs)	Time consumed by a scheme (number of operation/time consumed)		
		SMCS	SET [8]	MSET [8]
Asymmetric key encrypt.	2512.3	1/2512.3	7/17586.1	3/7536.9
Symmetric key encrypt (AES)	79.3	2/158.6	3/237.9	9/1427.4
Hash function	112.3	6/673.8	4/449.2	4/449.2
Exclusive-or operation	0.21	10/2.1	0/0	0/0
Binary addition	0.71	5/3.55	0/0	0/0
Total time		3350.35	17082.4	9413.5

Remark : (x/y stands for that a scheme requires the corresponding operation x times, consuming y μsec)

it provides in a wireless environment can be further enhanced. In our system, we have five authentication mechanisms, including authenticating OP_code, checking T_{nonce}, verifying HMAC(), verifying merchant and validating consumer. So we dare to say that the SMCS is more secure and faster than the other two compared schemes.

Chapter 5

Conclusions and Future Studies

A feasible convenient m-commerce system needs to be securely protected and conveniently performed both outdoors and indoors. In fact, the best way for m-commerce payment is paid by credit cards. The m-commerce system that integrates the two characteristics is truly a convenient m-commerce system. Of course, security is also an important characteristic of a m-commerce system. Generally, the SMCS has the three characteristics.

Before the consumer starts using this SMCS's APP for m-commerce, the consumer and card-issuing bank build the *DCC* and the APP to form a closed security mechanisms. After CA verifies the merchant's business registration certificate, it authenticates the merchant to ensure the legality and security of the merchant. Other important issue of the SMCS is that the card-issuing bank certificates the m-commerce payment request message sent by the consumer by using credit card's dynamic authentication code and the merchant's dynamic authentication code, so that the bank can confirm both the identities of the consumer and the merchant, thus ensuring the undeniable characteristics of this transaction. Theorem 1 indicates that messages transmitted in the SMCS are protected by a two-dimensional stream cipher technique. Theorem 2 shows that the SMCS can effectively protect the m-commerce payment request message against a variety of attacks. Theorem 3 further illustrates that if the m-commerce payment request message can pass all

the safety certification performed by the card-issuing bank, it must be sent by a valid consumer who uses his/her own mobile phone. Even though the consumer loses his/her mobile phone, the SMCS is still secure since the m-commerce activities through the mobile phone cannot be unauthorizedly performed, thus ensuring a high security level for the SMCS.

Our follow-up work is to build a complete SMCS simulation system, and look for possible cooperative banks to run the SMCS. The purpose is to make sure it is secure and feasible. We will also derive the reliability and behavior model for the SMCS so that users can predict its reliability and behaviors before using it. These constitute our future studies.

References

[1] Nabi F. Secure business application logic for e-commerce systems. In *Computers and Security*, volume 24, pages 208 – 217, 2005.

[2] Mahmoudi. N. and Duman. E. Detecting credit card fraud by modified fisher discriminant analysis. In *Expert Systems with Applications*, volume 42, pages 2510 – 2516, 2015.

[3] Steve Gold. The evolution of payment card fraud. In *Computer Fraud and Security*, volume 2014, pages 12 – 17, 2014.

[4] Oppliger. R, Hauser. R, and Basin. D. SSL/TLS session-aware user authentication revisited. In *Computers and Security*, volume 27, pages 64–70, 2008.

[5] Das. M.L and Samdaria. N. On the security of SSL/TLS-enabled applications. In *Applied Computing and Informatics*, volume 10, pages 68 – 81, 2014.

[6] Lu. S. and Smolka. S.A. Model checking the secure electronic transaction (SET) protocol. In *7th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, pages 358–364, 1999.

[7] Bella. G, Massacci. F, and Paulson L.C. Verifying the set registration protocols. In *IEEE Journal on Selected Areas in Communications*, volume 21, pages 77–87, 2003.

[8] Shedid. S.M and Kouta. M. Modified SET protocol for mobile payment: An empirical analysis. In *International Conference on Software Technology and Engineering*, volume 1, pages 350–355, 2010.

[9] Yong X. and Jindi L. Electronic payment system design based on SET and TTP. In *International Conference on E-Business and E-Government*, pages 275–278, 2010.

[10] Creditcards.com. `http://www.creditcards.com/`, 2015.

[11] Bicakci. K, Unal. D, Ascioglu. N, and Adalier. O. Mobile authentication secure against man-in-the-middle attacks. In *Procedia Computer Science*, volume 34, pages 323–329, 2014.

[12] Badra. M. and Urien. P. Toward ssl integration in sim smartcards. In *Wireless Communications and Networking Conference*, volume 2, pages 889–893, 2004.

[13] Bisel. L.D. The role of SSL in cybersecutiry. In *IT Professional*, volume 9, pages 22–25, 2007.

[14] Zhao H. and Liu R. A scheme to improve security of ssl. In *Pacific-Asia Conference on Circuits, Communications and Systems,*, pages 401–404, 2009.

[15] Du. L, Hu. X, Li. Y, and Zhao G. A csk based SSL handshake protocol. In *IEEE International Conference on Network Infrastructure and Digital Content*, pages 600–603, 2009.

[16] Petridou. S and Basagiannis. S. Towards energy consumption evaluation of the SSL handshake protocol in mobile communications. In *Annual Conference on Wireless On-demand Network Systems and Services*, pages 135–138, 2012.

[17] Venkataiahgari. A.K, Atwood. J.W, and Debbabi. M. Secure e-commerce transactions for multicast services. In *IEEE International Conference on and Enterprise E-Commerce Technology*, page 18, 2006.

[18] Guan. H.J. The research of set-based electronic payment system model. In *International Conference on E-Business and Information System Security*, pages 1–4, 2009.

[19] Sherif. M.H, Serhrouchni. A, Gaid. A.Y, and Farazmandnia. F. Set and ssl: electronic payments on the internet. In *IEEE Symposium on Computers and Communications,*, pages 353–358, 1998.

[20] Chaudhary. A, Ahmad. K, and Rizvi. M.A. E-commerce security through asymmetric key algorithm. In *International Conference Communication Systems and Network Technologies*, pages 776–781, 2014.

[21] Huang. Y.L., Leu. F.Y., and Wei. K.C. A secure communication over wireless environments by using a data connection core. In *Mathematical and Computer Modelling*, volume 58, pages 1459 – 1474, 2013.

[22] Huang. Y.L., Dai. C.R., Leu. F.Y., and You. I. A secure data encryption method employing a sequential-logic style mechanism for a cloud system. In *International Journal of Web and Grid Services*, volume 11, pages 102–124, 2015.

[23] Phone Lin, Hung yueh Chen, Yuguang Fang, Jeu-Yih Jeng, and Fang sun Lu. A secure mobile electronic payment architecture platform for wireless mobile networks. volume 7, pages 2705–2713, 2008.

[24] Hwang. M.S., Lin. I.C., and Li. L.H. A simple micro-payment scheme. In *Journal of Systems and Software*, volume 55, pages 221 – 229, 2001.

[25] Li. Y. and Zhang. X. Securing credit card transactions with one-time payment scheme. In *Electronic Commerce Research and Applications*, volume 4, pages 413 – 426, 2005.

[26] Naqvi. S.I and Akram. A. Pseudo-random key generation for secure HMAC-MD5. In *IEEE International Conference on Communication Software and Networks*, pages 573–577, 2011.

[27] Najjar. M and Najjar. F. d-hmac dynamic hmac function. In *International Conference on Dependability of Computer Systems*, pages 119–126, 2006.

[28] Leu. F.Y., Huang. Y.F., and Chiu C.H. Improving security levels of ieee802.16e authentication by involving diffie-hellman PKDS. In *Universal Computer Science*, volume vol. 17, pages 891–911, 2010.

[29] Haung. Y.L., Leu. F.Y., Chenand. J.H., and Chu William C.C. A true random-number encryption method employing block cipher and PRNG. In

Computer Science and Information Systems, volume vol. 11, pages 905 – 924, 2014.