Secure One-way Mobile Payment System Keeping Low Computation in Mobile Devices

Wooseok Ham[1], Hyungki Choi[1], Yan Xie[1], Misung Lee[2], and Kwangjo Kim[1]

[1] International Research center for Information Security (IRIS)
Information and Communications University (ICU)
58-4 Hwaam-dong, Yusong-gu, Daejeon, 305-732, S. Korea
{tarzan92, hkchoi, yanxie,kkj}@icu.ac.kr
[2] School of Management
Information and Communications University (ICU)
sensy74@icu.ac.kr

Abstract. In this paper, we present a secure one-way mobile payment system that executes only two modular multiplications, one modular inverse and two hashings by the customer using two public key pairs and keyed hash function. Thus the customer(*i.e.* mobile device) conserves low computation load without any expensive modular exponentiation required for RSA or Diffie-Hellman operation. In addition, as the customer need not participate in the deposit phase, only one unilateral communication from the customer to the vendor is sufficient to complete payment. These characteristics will make our scheme easily applicable to the mobile environments. The security of the proposed mobile payment system is proved to be equivalent to the intractability of the discrete logarithm problem.

1 Introduction

With the widely spreading of mobile devices, M-commerce keeps growing popularity. A lot of applications can be imagined for M-commerce such as banking, trading, shopping, lottery, and game. But, all these applications has a common attribute: the consumer has to *pay* for services or products. Therefore, the electronic payment system will play an inevitable role in making M-commerce to be popular.

Since the mobile device first introduced in the world, there has been rapid development of new functions, improvement of services, and the enhancement of the computing power of mobile devices make M-commerce more profitable and promising.

But, there is still antipathy from the public to buy products or services online and pay for them also on the Internet. The main problem is that almost all Internet users are aware of credit card fraud committed by hackers during transmission over the communication channel. Therefore, on-line mobile service provider has to figure out how to pay out without any dispute through every commercial transaction and even, when disputes take place, how the system

to resolve them without losing fairness. Similarly, mobile payments face with a number of problems from not only performance but also security points of view. As a result, mobile payments also become an emerging issue in the mobile commerce security.

When we try to design a mobile payment system, the first thing we have to consider is the difference between the mobile and other on-line environments; the computational power of main processor, the quality of service, and the power consumption, *etc.*, which mobile devices must provide. Although mobile devices have been evolved to the highly computing resource, computational power and network quality are still too limited to adopt the existing electronic payment systems that need several exponentiations to complete payment procedures.

As a result, mobile devices are inevitably constrained with respect to cryptographic functions and the mobile(wireless) communication is highly vulnerable to various attacks since eavesdropping is executed easily than the wired channel in general. Several attempts have been tried to support high security in the mobile networks such as WTLS[17] under WPKI, M-VPN[3], and SSL in i-mode[5]. But they have still many arguments with respect to security and performance.

To the best of our knowledge, two approaches have been done to the mobile payment systems up to date. One is to exploit the properties of mobile agent to reduce the computational overhead by a customer. The other is to make use of mobile devices as an authentication tool.

By Mobile Agent. Since mobile devices possess limited communicational and computational capacity, mobile payment systems proposed in [8, 13, 18] utilize the idea of "mobile agent" which has autonomity and migration capability. The schemes in [13, 18], employing mobile agent techniques, has accommodated SET[15] protocol in which several public key computations are followed for payment. On behalf of a customer, the mobile agent performs all processes necessary in SET with the customer's confidential data. Lee *et al.*[8] also made use of SET but combined it with *Millicent*[9] in order to make the micropayment system available in mobile devices.

However, the critical problem is the prevention against a malicious host. To execute a purchase transaction, payment agent has to bring the customer's confidential data to the designated host. A random symmetric key used to encrypt payment instruction is generated in the merchant host. Thus, as all computations like encryption and signature verification are performed in the merchant host, we have to guarantee the correctness of computation and confidentiality of customer information which the mobile agent accompanies. Although a few methods like [7, 14] to protect the mobile agent have been initiated, they increase the total computation adopting additional functions such as proxy signature or proxy certificate. In fact, no known general solution exist against a malicious host.

By Mobile Device. Instead of using the mobile agent, a different approach has been introduced to the financial industry for the mobile payment systems such as Paybox.net[11] in Germany and Mobilix[10] in Denmark, *etc.* They simply use the mobile device as an authentication tool to confirm customer's payment information and approval by sending secret short code(*e.g.* password) over the air.

Even though they support customer's identification through his/her mobile phone number, the session among whole participating parties such as the financial facility, the vendor and the customer should be kept on-line during the transaction to check the validity and fairness of the payment. Without customer's prompt confirmation, the transaction will not able to be completed. Furthermore, the inherent problem of the mobile networks, the existence of mobile gateway, can be exploited to obtain customer's private information like account number or password by an attacker.

Our Approaches. There are various operational models for M-commerce composed of three entities; the customer(C), the vendor(V), and the bank(B). However we specify the connections among three entities for our scheme as Fiq.1.

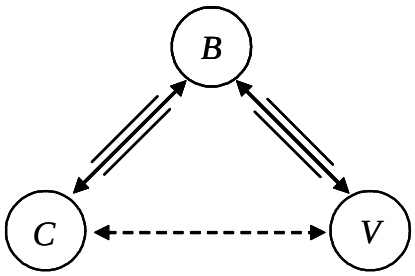

Fig. 1. Operational model

Here, note that only the connection between the customer and the vendor(*i.e.* purchase) is set up through the wireless channel denoted as the dotted arrow. The customer has a mobile device like a mobile phone and the vendor provides corresponding mobile services.

When the customer or the vendor interacts with the bank, sensitive information such as account information or password and even money itself are transferred into the other end. In that case, a reliable and secure connection against passive and active attacks is mandatory. So, the other two connections(*i.e.* withdrawal and deposit) depicted as the solid arrows are assumed to be established through the *secure* wired channel by using the well-known security protocol like SSL[16]. In consequence, we mainly focus on the design of the mobile payment system which is secure against various attacks under the wireless network between the customer and the vendor.

Our approach on the payment is to utilize *off-line digital money* directly, which means the bank need not be *on-line* in the payment processing for goods under the above operational model. This is different from the two previous approaches where the bank(or the financial facility) should be always *on-line* to mediate the payment. In our scheme, we use a public key cryptosystem based on the discrete logarithm problem(hereinafter, DLP) and a keyed hash function without adopting a mobile agent technique. However, the computation load of the customer is distinctively small: *two modular multiplications, one modular inverse and two hashings* only. The number of communication is just *one* from the customer to the vendor. Consequently, these characteristics make our payment system applicable in the mobile environments. We show that the security of our system is equivalent to the intractability of the DLP.

Organization. The rest of the paper is organized as follows: Section 2 introduces general considerations in the mobile payment systems such as its inherent limitations and security requirements. In Section 3, we propose our scheme consisting of Withdrawal, Purchase, and Deposit protocols. In Section 4, the security aspects of our scheme are discussed and also the overall performance of the scheme is evaluated in terms of computation and communication. We end with our conclusions in Section 5.

2 Considerations

Before getting into our proposed scheme, we want to provide some basic concerns which we try to solve and overcome in the mobile environment. The general architecture of the mobile communication will be omitted here. For the interested readers, refer to [17]. At first, the constraint of the mobile environments will be provided for readers to recognize why the current hot issues with respect to the mobile communication have started. In addition, we will describe electronic payment requirements for the mobile payment systems to be accomplished.

2.1 Limitations

We will look into the general security and performance limitations in the mobile environments more closely to recognize what we can or can't do with mobile devices.

Security Limitations

Most severe problem in security of the mobile networks comes from the inherent property of using proxy, *i.e.* mobile gateway, to communicate. Conventional mobile communication such as WAP[17] use two-tier transport layer: user and proxy, proxy and service provider, because each tier follows different protocol, the proxy always plays as a converter in transferring messages to fit on each protocol thus it can get full information on the messages. Thus we need strong

trust on that this proxy works properly in secure way to transfer data through the wireless networks. Furthermore, the proxy should not save data in transaction to its storage or memory and must guarantee that authority only can get access to manage it.

Another problem arises when public key cryptosystems, based on exponent computation, are used for the security and availability. For example, in order to assure security against cryptonalysis and various attacks, RSA must use 1024-bit modulus. This will be a big burden on mobile devices with respect to computation as they have limited resources.

Performance Limitations

Although wireless businesses are gaining popularity more and more, there exist performance limitations comparing to desktop environment that severely restricting what we can do in the mobile payment systems. In terms of hardware devices, the widely acknowledged performance limitations[17] are: *low CPU power, less memory (ROM and RAM), restricted power consumption, small display, and a variety of different input devices, etc.*

The problem is how we can transform currently available applications in the wired environments into the wireless environments without degrading the underlying properties. To make more convenient use of mobile devices, the size of a mobile device is getting smaller. Therefore, the available resources in desktop environment are not equally given when designing or implementing in the mobile environment. Obviously, above limitations prohibit mobile devices from heavy computations to require for conventional electronic commerce applications like SET. Furthermore, the serious concern is how to work well with a lot of different mobile devices.

In addition, forwarding message requires reliable network bandwidth. But we also have network limitations with the mobile communications[17] such as: *less bandwidth, more latency, unstable connection, and high error rate.*

2.2 Electronic Payment Requirements

Here, we classifies the basic security and performance requirements[4] of the electronic payment systems that must be fulfilled in the mobile payment systems as follows:

Unforgeability. Only authorized entity can issue coins.
Double-spending prevention. One issued coin cannot be used more than once.
Efficiency. The system must be efficient in terms of storage, communication, and computation.

Although there are many other requirements from the viewpoint of security and availability like anonymity, atomicity, divisibility, unlinkability, and transferability, *etc.*, the requirements to be addressed rely on the specific situation

where the payment system runs. In our contribution, we design a mobile payment system satisfy only the general requirements listed above as a first step.

As in the real life, there must be a way of controlling money. If anyone can mint money freely, it's totally unnecessary for us to secure the money because if you need money, you can make it whenever you want. In electronic payment, more attention should be paid for the fact that digital script can be easily copied. Therefore, only authorized entity (*i.e.* Bank) should issue money and the issued money should be hard to counterfeit.

In electronic payment system, there must a manner to prevent from spending the same money script more than once. If this happens, a lot of disputes will occur in the payment system and the bank can be cheated by the customer or the vendor. In fact, double spending is a weak forgery of the payment script. Preventing double spending is very important in the electronic payment system.

From the performance limitations described earlier, reducing computation and communication load without compromising other existent characteristics is necessary in the generic mobile environments. We also focus on providing better efficiency with basic security requirements that listed above in constructing payment systems.

3 Our Proposed Scheme

3.1 Notations

Our protocol is made up with three entities: the customer(C), the vendor(V), and the bank(B) as depicted in Fiq.1. $\mathbb{Z}_p^*$ denotes a multiplicative group of integers modulo p. p and q are large primes such that $q|p-1$ and solving discrete logarithm problem is hard in the unique subgroup $G_q \subset \mathbb{Z}_p^*$ of order q. g is a generator of G_q. $a \in_R G_q$ means that a is chosen at random from G_q according to the uniform distribution and a^{-1} denotes the multiplicative inverse of a. In our scheme, all exponentiation is computed in modulo p and addition and multiplication are done in modulo q, unless otherwise specified. The symbol $\|$ denotes the concatenation of two strings of group elements. We represent a collision resistant one-way hash function [2] as $\mathcal{H} : \{0,1\}^* \longrightarrow \{0,1\}^l$ $(l \geq 160)$. KH denotes keyed hash chain. PD is a public domain like a bulletin board which is controlled by B. Anyone can access to PD and retrieve public information from it. MD denotes mobile devices such as a mobile phone and PDA.

3.2 Building Blocks

Before describing our main protocols, we introduce the basic building blocks. First, we assume that each customer has two private and public key pairs, $(x_1, y_1(= g^{x_1}))$ and $(x_2, y_2(= g^{x_2}))$ and public keys are opened to the public. These two public keys makes our proposed scheme secure sustaining less computation. Generation and distribution of these key pairs are the same with the usual public key cryptosystems except one has two public keys. In addition, we take use of a distinct hash chain technique, *Keyed Hash Chain* as a building block.

Keyed Hash Chain. To assign the serial number on the current protocol run which guarantees the uniqueness of transaction, our scheme exploits keyed hash chain technique. Each output of the keyed hash chain works as *nonce*, which is used for ascertaining uniqueness of the current transaction in our scheme. Based on the difficulty of computing the inverse of hash function, the unique serial number is able to be generated by keeping the key used in the keyed hash chain in secret. Unlike reversed hash chain technique used in [6, 12], which necessarily calculate n hashings to make use of n th output, our keyed hash chain outputs are used in increasing order by which we can save memory and computation load.

$$\begin{aligned} \text{Generation}: \quad & KH_0 = k, \\ & KH_i = \mathcal{H}(k \| KH_{i-1}) \text{ where } i = 1, \ldots, l. \\ \text{Usage}: \quad & \text{Increasing order (from 1 to } l). \end{aligned}$$

3.3 Detailed Protocols

Our scheme consists of three protocols: **Withdrawal**, **Purchase**, and **Deposit**. Before describing the entire protocol, note that both Withdrawal and Deposit protocols are performed over a secure wired channel and only Purchase protocol is done through the mobile networks as stated in Section 1. Each protocol has the following main functions:

Withdrawal (C $\leftrightarrow$ B) : C requests setting-up for the mobile payment to B, who decides and publishes initial value observing the predetermined rule and issues the confirmation receipt on the request.

Purchase (C $\leftrightarrow$ V) : C constructs the payment script as much as the negotiated price of a product and pays it to V, who verifies the correctness of the payment script.

Deposit (V $\leftrightarrow$ B) : V asks deposit on the payment script received from C then B reimburses the appropriate amount of money to V after verification.

Before our proposed scheme starts, we assume that two public keys of C, y_1 and y_2 have published in advance to the public key repository which permits anyone to access and retrieve the relevant information. The corresponding secret keys, x_1 and x_2 are preserved by C in secret.

Protocol 1 : Withdrawal

C sends initiative request for setting up the mobile payment to B with his identity ID_C, account information AI and additional information ε to confirm C's account and identity. If C's account is good then B selects a random number k from the subgroup G_q and calculates y_k. B associates k with C's account and keeps it in safe. This value will be used for the verification of the payment script

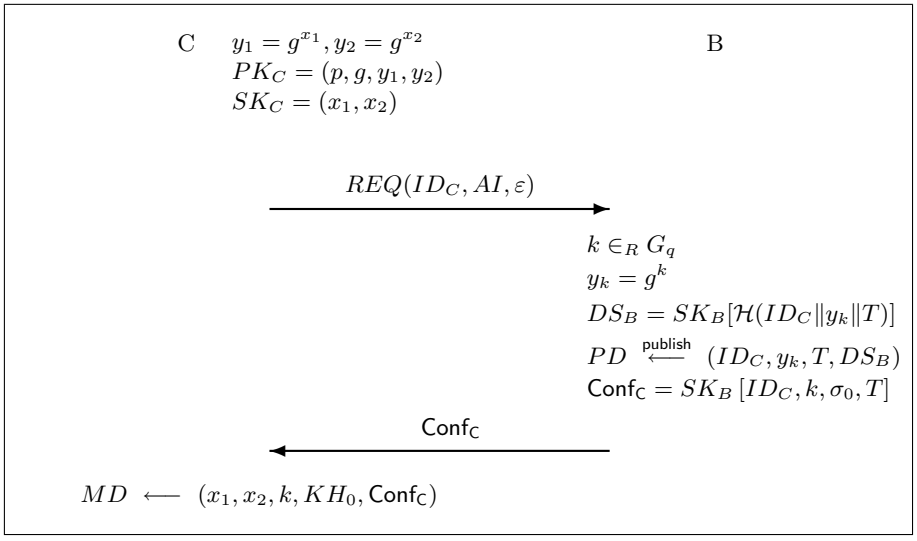

Fig. 2. Withdrawal

from V in Deposit protocol and guarantee the prevention of double spending. B also decides the limitation of withdrawal σ_0 subject to the condition of C's account including some additional credit-loan. B publishes ID_C and y_k with T, issue and duration time of k, to PD so they become publicly accessible. Duration time is predetermined among participating entities. Whenever B releases the data to PD, for the integrity of PD, he generates digital signature DS_B on the data and posts to PD to convince who gets the data.

B generates a confirmation receipt $\mathsf{Conf_C}$ on the received values using his secret key SK_B and returns $\mathsf{Conf_C}$ as the receipt to C through the secure wired channel. $\mathsf{Conf_C}$ makes B not to repudiate his agreement on the initiation of the mobile payment with C and the correctness of k. In fact, $\mathsf{Conf_C}$ is not used any more in other protocols. Its role is just an evidence about the generation of k when disputes happens among entities. C decrypts $\mathsf{Conf_C}$ with PK_B and checks the lifespan of committed k and the available amount of money σ_0. Then C stores securely his own secret keys x_1 and x_2, committed values k, initial keyed hashing output $KH_0 (= \mathcal{H}(k))$, and $\mathsf{Conf_C}$ to his mobile device.

For the security enhancement, we may employ smart card which has the tamper-proof property as an alternative to store those confidential data. In terms of anonymity, we can substitute ID_C for pseudonym which is issued by B, for instance, in the similar way that generates KH. B and C perform Withdrawal protocol whenever the following cases take place:

- The lifetime of k expires or the value of k is compromised.
- The summation of payments exceeds σ_0.
- Public key pair of B should be regenerated.

– C wants to alternate his account.

Protocol 2 : Purchase

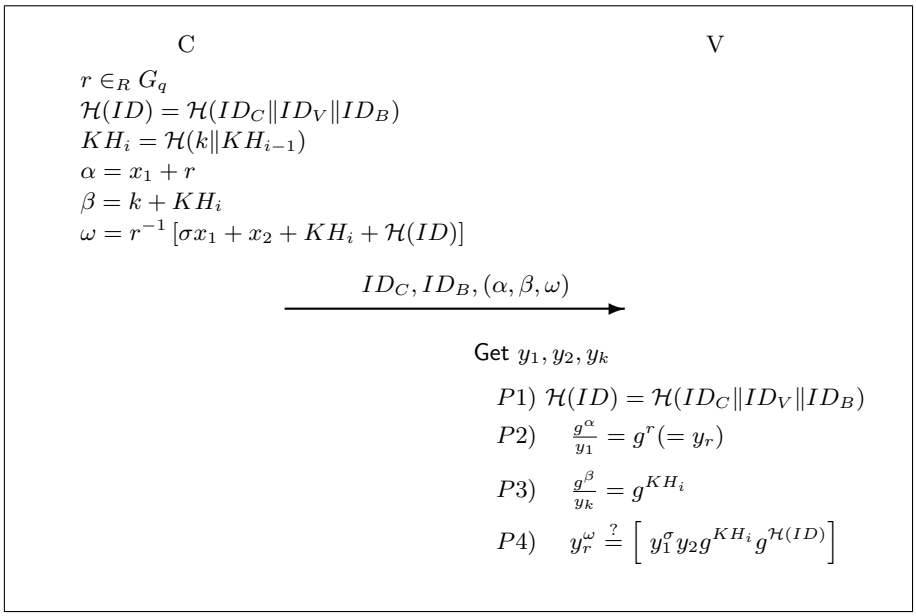

Fig. 3. Purchase

This protocol is carried out between C and V over the wireless channel. Assume that the negotiated price of a product is σ which denotes the i-th payment of C for the sake of simplicity. For constructing payment script, C first picks up r at random from the subgroup G_q and hashes the concatenated identities of C, V, and B resulting in $\mathcal{H}(ID)$. Then C calculates keyed hash chain output KH_i that indicates the i-th payment and yields α, β, and ω as in Fig.3. C sends ID_C, ID_B, and (α, β, ω) as payment to V. By using two public keys and attaching a random number to the message, these messages reveal no information on secrets x_1 and x_2. We will discuss this more later.

After receiving the payment script, V gets y_1 and y_2 from the public key repository and the public value y_k from PD using ID_C. V proceeds equations from $P1$ to $P4$ in order to verify the correctness of the received payment script. First, V computes $\mathcal{H}(ID)$ and then extracts $g^r (= y_r)$ from α and g^{KH_i} from β, using y_1 and y_k , respectively. V finally performs equation $P4$ to ascertain the payment script representing the price σ. If it passes, V keeps ID_C and g^{KH_i} for the period of lifespan of k to prevent double spending of the script. When V

finds the same g^{KH_i}, by comparing with the stored values, he denies the payment script and closes the protocol. Thus C can't buy any goods of V with the same script due to the KH_i in β. The verification process, equation $P4$, could be justified as follows:

$$\begin{aligned} y_r^{\omega} &= g^{rr^{-1}[\sigma x_1 + x_2 + KH_i + \mathcal{H}(ID)]} \\ &= g^{\sigma x_1 + x_2 + KH_i + \mathcal{H}(ID)} \\ &= y_1^{\sigma} y_2 g^{KH_i} g^{\mathcal{H}(ID)}. \end{aligned}$$

If this fails, V may claims the payment script is invalid or forged and terminates the transaction. Notice that only the person who knows x_1, x_2, and k can compose legitimate messages α, β, and ω. In addition, since the hashed value of identities $\mathcal{H}(ID)$ is included in ω, even though an attacker intercepts the message, he cannot request deposit on the payment script and cannot insist that the script belongs to himself.

Protocol 3 : Deposit

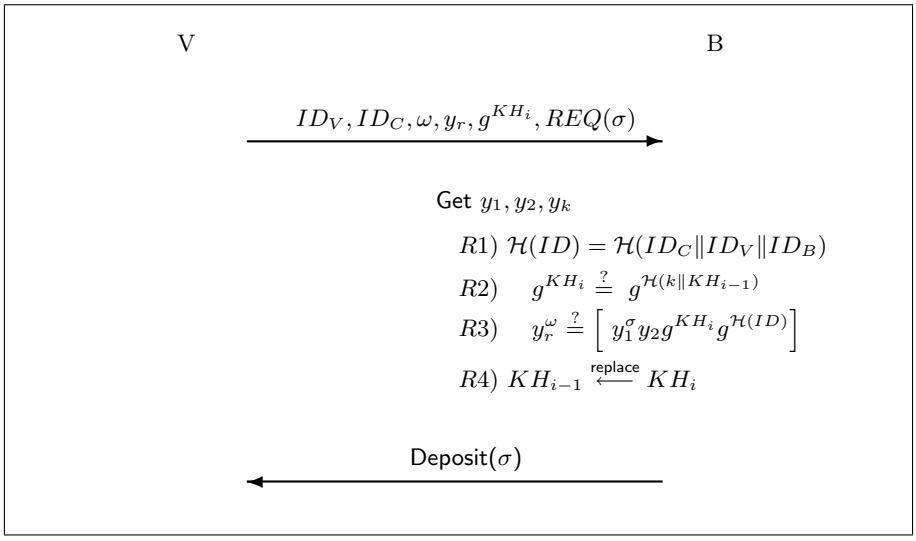

Fig. 4. Deposit

Deposit protocol occurs between V and B through a secure wired channel. In order to get refund on the payment script, V needs to be authenticated by B that he is indeed the right merchant who sold his goods to C and received the valid payment script from the designated C. V sends the request tuple, (ID_V, ID_C, ω, y_r, g^{KH_i}, $REQ(\sigma)$), to B. B gets C's public keys (y_1, y_2) and retrieves

k and KH_{i-1} corresponding to the ID_C's account which is maintained by B in safe.

B yields $\mathcal{H}(ID)$ and KH_i then confirms the transferred message through equations $R2$ and $R3$. First, B certifies the uniqueness of the script by equation $R2$ with calculated KH_i then checks the correctness of the payment script by equation $R4$. We omit justification of equation $R4$ because it is the same with $P4$ in Purchase protocol. If equations $R2$ and $R3$ are held, B replaces the value of KH_{i-1} with KH_i for the next verification. Finally, B reimburses the appropriate amount of money $\mathsf{Deposit}(\sigma)$ back to V. During the confirmation, if the accumulated of money spent by C exceeds σ_0, B eliminates g^k of C from PD then C must initiate Withdrawal protocol again to pay further payment.

4 Evaluation

We evaluate our scheme with respect to security and performance in detail.

4.1 Security Evaluation

We may think three types of attack to forge the payment script due to the combination of entities: Vendor-only, Bank-only, and Vendor-Bank-together. Among them, the most powerful condition is the collusion of Vendor and Bank since they can obtain all messages during the transaction. So, we only consider this type of attack here.

Unforgeability. To prove unforgeability of the payment script, we use the following lemma:

Lemma 1 *If solving the discrete logarithm problem is hard under a group, for the given pair (g^x, μ) where x is the secret key and g^x is the public key, finding random element τ of the group satisfying such that $\mu = x + \tau$ is equivalent to the difficulty of the discrete logarithm problem.*

Proof. It is straightforward to show the proof. Since we don't know the value of x from g^x by the intractability of DLP under a group which is solving DLP is hard in the polynomial time and τ is a random group element, the only way to get τ is to find the discrete logarithm of (g^μ/g^x) such that $g^\tau = (g^\mu/g^x)$. It leads to solve DLP again, so we prove Lemma 1. ■

In case of the collusion of V and B, β is meaningless from the view of B. So, we keep an eye on the values α and ω. V and B have knowledge on σ, KH_i, and $H(ID)$ from the received messages and the keeping values k and KH_{i-1} to B. However, given α, they hardly obtain the random number r by Lemma 1. In addition, they don't know $(\sigma x_1 + x_2)$ as it is difficult to get x_1 and x_2 due to DLP. Without knowing r, it is impossible to compute the inverse r^{-1} which is multiplied to each term in ω. Therefore V and B cannot generate another payment script from the messages.

Here, the reason why we make use of two public key pairs to improve security becomes obvious. Assume that we use one public key pair (x_1, y_1). By changing of variable, x_1 into $(\alpha - r)$ in ω, B is able to get r^{-1} since B knows all other values; $\alpha, \omega, \sigma, KH_i,$ and $\mathcal{H}(ID)$. It leads to the disclosure of the secret key x_1, since it is simple to know r and x_1 from r^{-1}.

Although V and B have plural payment scripts on the same amount of money σ from the identical customer, they only can get the following values:

$$\alpha - \alpha' = r - r'.$$
$$\omega - \omega' = (r^{-1} - r'^{-1})(\sigma x_1 + x_2 + KH_i + \mathcal{H}(ID)).$$

With the value of subtraction of α and α', they cannot induce r or r', which are chosen at random from the subgroup G_q, and cannot discover any helpful relationship between $(r - r')$ and $(r^{-1} - r'^{-1})$. The unique way to know both r and r' is to calculate the discrete logarithm of g^r or $g^{r'}$ on each.

As a result, there is no way to generate an eligible payment script by V and B without changing α and ω, even though V and B conspire together, as far as resolving DLP is intractable.

Double Spending. The result of keyed hash chain KH_i makes sure the uniqueness of the current payment protocol run. Only the customer, who has k which comes from B in a secure way, can compute eligible KH_i by the underlying property of keyed hash function. So, when the same g^{KH_i} is returned to the bank, the bank can easily identify double spending. Furthermore, since only the owner of public key pairs y_1 and y_2 can make a legitimate payment script, the customer cannot repudiate double spending of the payment script when the bank informs the customer of the fact of double spending. One-wayness of hash function also makes an attacker is unable to find the pre-image of the result of hashing.

From these discussions, we get the following theorem:

Theorem 1 *Our scheme satisfies the basic security requirements; unforgeability and double spending prevention, of the mobile payment system.* ■

Remark. We are able to imagine various other attacks by the participating parties themselves to cheat other parties. Money forgery by any party is unattainable as described previously except the customer who is in the possession of secret keys (x_1, x_2).

The customer may try to send his same payment script to other vendors or even the same vendor to other goods. But both cases are not permitted. Since ID of the vendor who sold the goods to the customer is included in ω, the same payment script fails to the verification process by the other vendor in the former case. The latter case is also easily detected as the vendor is supposed to hold ID_C and g^{KH_i} during the lifespan of y_k that is certified by observing the bank's publication.

We might imagine another possibility of only substituting ID_V to other vendor's ID preserving the other values as the same by the customer. However this misbehavior will be detected by the bank from the index KH_i during Deposit protocol and the vendor who received invalid payment script will not be paid. In that case, the vendor can accuse the customer of cheating in a court of law or to an arbitrator with the payment script as an evidence.

We skip the details on security against the other possible attacks by the vendor or the bank since the proposed system can protect or solve with similar reasons above without losing confidentiality and fairness of the payment script.

4.2 Performance Evaluation

We discuss the performance of our scheme in terms of computation and communication which are main interests in the mobile applications.

Computation. In general, B and V are supposed to have enough powerful computational resources to execute several exponentiations. So we only take into consideration on the C's computational capability. Since Withdrawal protocol is carried out through the wired channel and C can use high-performance computation equipment like personal computer for withdrawal, one exponentiation to verify $\mathsf{Conf_C}$ is not expensive. During Purchase protocol, C executes only *two modular multiplications, one modular inverse, and two hashings* to constitute the payment script α, β and ω(see Fig.3). As addition operation is negligible, we don't take into account on it. Furthermore, C need not participate in Deposit protocol again. As a result, the total computation, two modular multiplications, one modular inverse, and two hashings are so reasonable that we are able to operate several times on mobile devices with little consumption of power and memory and less CPU power.

Communication. Withdrawal and Deposit protocols are executed through the wired channel fewer than Purchase protocol. Thus let's consider only the communication taking place in Purchase protocol. Suppose that the size of each party's ID is 20 bit and q is 160 bit and SHA-1 is used for the hashing. When a customer pays out to a vendor in Purchase protocol, the customer sends the following message to the vendor as shown in Fig.3:

$$ID_C, ID_B, (\alpha, \beta, \omega).$$

The total size of the message is

$$20 + 20 + (3 \times 160) = 520 \text{ bit},$$

since α, β, and ω are computed on modulo q. Message transfer from the customer to the vendor occurs only *once* during Purchase protocol. As a result, the total size, *520 bit*, is quite reasonable to transmit through the wireless networks with the limited bandwidth.

5 Conclusions

We suggested a secure and efficient mobile payment system that calculates only two modular multiplications, one modular inverse, and two hash computations, using two public key pairs and keyed hash function. Thus the customer(*i.e* mobile device) conserves low computation exempting from any modular exponent computation which is generally used in other electronic payment systems. Even one unilateral small message transfer from the customer to the vendor is sufficient to complete payment. This characteristic is very effective in the wireless networks with the limited bandwidth. The security of our scheme is based on the intractability of the discrete logarithm problem and the one-wayness of hash function.

We state that our scheme can be easily applicable for M-commerce due to the low communication, the light computation, and simple composition of protocol. As further works, it will be valuable to design an *anonymous fair off-line* mobile payment system. Also, devising a mobile payment system with minimal complexity will be a good challenge when all parties C, B, and V are in the wireless channel.

Acknowledgements

The authors are deeply grateful to the anonymous reviewers for their valuable suggestions and comments on the first version of this paper.

References

1. C. Kaufman, R. Perlman, and M. Speciner, Network Security PRIVATE Communication in a PUBLIC World, *Prentice Hall*, New Jersey, NJ, 1995.
2. I.B.Damgård, Collision Free Hash Functions and Public Key Signature Schemes, *Advances in Cryptology–Eurocrypt'87*, LNCS304, Springer-Verlag, pp.203–216, 1988.
3. ERICSSON, Mobile Virtual Private Network, available at http://www.ericsson.com.
4. H. Kim, M. Seo, J. Baek, and K. Kim, Requirements and Comparision of the Existing Electronic Cash Protocols, *Proc. of WISC99*, pp.231–251, 1999.
5. NTT DoCoMo, available at http://www.nttdocomo.com.
6. L. Lamport, Password Authentication with Insecure Communication, *Communications of the ACM*, 24(11), pp.770–772, 1981.
7. B. Lee, H. Kim, and K. Kim, Secure Mobile Agent using Strong Non-designated Proxy Signature, *ACISP2001*, LNCS 2119, Springer Verlag, pp.474-486, 2001.
8. T. Lee, Y. Yip, C. Tsang, and K. Ng, An Agent-Based Micropayment System for E-Commerce, *E-commerce Agents*, LNAI 2033, Springer-Verlag, pp.247–263, 2001.
9. M.S.Manasse, The Millicent Protocol for Electronic Commerce, *Proc.of the 1st USENIX Workshop on Electronic Commerce*, 1995. available from http://www.millicent.org/works/details/papers/mcentny.htm.

10. MobiliX, available at http://mobilix.org.
11. Paybox.net, available at http://www.paybox.net.
12. R.Rivest and A.Shamir, PayWord and MicroMint: Two Simple Micropayment Schemes, *International Workshop on Security Protocols*, LNCS 1189, Springer-Verlag, pp.69–87, 1996.
13. A. Romao, and M. M. da Silva, An Agent-Based Secure Internet Payment System for Mobile Computing, *TREC98*, LNCS 1402, Springer-Verlag, pp.80–93, 1998.
14. A. Romao, and M. M. da Silva, Secure Mobile Agent Digital Signatures with Proxy Certificates, *E-Commerce Agents*, LNAI 2033, Springer-Verlag, pp.206-200, 2001.
15. The SET Standard Book 1 Business Description, SETCo, available from http://www.setco.org.
16. Netscape Communications, The SSL Protocol, Version 3.0, available at http://wp.netscape.com/eng/ssl3/ssl-toc.html.
17. WAP Forum, available at http://www.wapforum.org.
18. X. F. Wang, K. Y. Lan, and X. Yi, Secure Agent-Meditated Mobile Payment, *PRIMA98*, LNAI 1599, Springer-Verlag, pp 162–173, 1999.
19. K. Wrona, M. Schuba, and G. Zabagli, Mobile payments-State of Art and Open problems, *Welcome2001*, LNCS 2232, Springer-Verlag, pp 88–100, 2001.