Off-Line Micro Payment Scheme with Dual Signature

Lih-Chyau Wuu* Kuang-Yi Chen Chih-Ming Lin

Institute of Computer Science and Information Engineering

National Yunlin University of Science & Technology

Yunlin 640, Taiwan, ROC

wuulc@yuntech.edu.tw

Received 5 November 2007; Revised 25 December 2007; Accepted 11January 2008

Abstract. In this paper we propose a secure and efficient off-line micro payment scheme. There are five parties in our scheme: Consumer, Bank, Merchant, Issuer and Trusted Authority. The coins are made by Bank in cooperation with Issuer to prevent not only Bank from impersonating a consumer to steal consumer's money but also Issuer from making coins of his own accord. The representation of coins is based on hash chain technique that the coin verification can be done quickly during a transaction. Furthermore, we propose a dual signature scheme extended from RSA algorithm that a legitimate coin must be signed by both Issuer and Consumer to assure no one except Consumer can spend coins, and no coin can be expanded from any legitimate coin chain. In summary, our system has the following properties: (1) Unforgeability and Unexpandability of coins. (2) Efficiency of coins verification. (3) No-usurpation of coins. (4) Anonymity of Consumer. (5) Traceability of double-spender.

Keywords: micro-payment, hash chain technique, RSA, double signature

1 Introduction

With the popularity of Internet, e-commerce has been discussed widely from both industry and academia [1-11]. Since Internet is an open network, providing a secure e-payment environment is fundamental to success of e-commerce. The simplest e-payment system involves three parties: consumer, merchant and bank. According to whether the bank is involved during a transaction, there are online e-payment systems [12-18] and offline e-payment systems [19,20].

In an online e-payment system, bank must verify the legitimacy of the coins received by merchant during payment time. This has resulted in an inefficient and inconvenient transaction manner. As for an offline e-payment system, since bank needs not to be involved during payment time, it is more difficult to protect against double spending or forging of coins than online e-payment systems.

In this paper, we propose a secure and efficient off-line micro payment scheme to have the following properties:

- **Unforgeability**: Only authorized organizations should be able to make coins.
- **Unexpandability**: No coin can be produced from any legitimate coin.
- **Anonymity**: The coins should be anonymous to protect the privacy of owners.
- **No framing**: Only the owner can spend the coin.
- **Double-spending prevention or detection**: If the same coin is used more than once, then it should be detected and be traced back to the double-spender.
- **Efficiency**: The computation burden added to all involved parties should be minimal.

The remainder of this paper is organized as follows. In Section 2, we describe our micro payment system architecture and give an overview of our scheme. Section 3 introduces the proposed dual signature method. Section 4 presents our micro payment protocol in detail. In Section 5, we analyze the security of the proposed scheme. Finally, Section 6 gives our conclusion.

* Correspondence author

2 System Architecture

The participants of our system include *Consumer*, *Bank*, *Merchant*, *Issuer* and a *Trusted Authority* as shown in Fig. 1. It is assumed that each participant has a certificate containing his/her real identity and long-term public key. Our system consists of five phases: register, withdraw, mintage, payment and deposit phases.

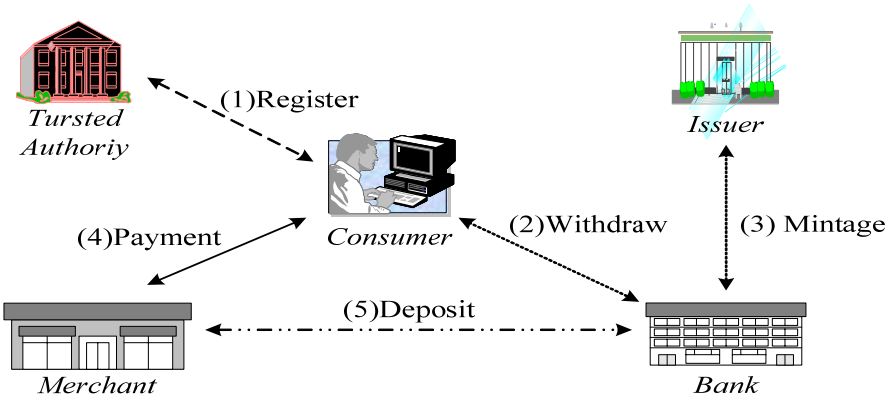

Fig. 1. The relationships among the participants.

Consumers must register with the Trusted Authority to obtain a Certificate containing a pseudo-identity and a corresponding public key. When a consumer applies a withdrawal request, the bank first authenticates the consumer and checks his/her bank balance, and then cooperates with the Issuer to make coins and generate three keys (K_{SI}, K_{SC}, K_P). A coin is signed by the Issuer with key K_{SI} during mintage phase, and signed by the Consumer with key K_{SC} during payment phase. Such a signature is called as *dual signature*. The Merchant uses the key K_P to validate the coin. In the deposit phase, *Bank* validates coins deposited by Merchant, and check whether the coins have been spent before. If detecting double-spending, *Bank* will ask the assistance of *Trusted Authority* to find out the real identity of the coin owner.

3 Dual Signature

The proposed dual signature scheme is extended from RSA algorithm. It requests that a valid coin must be signed by both the *Issuer* and the coin owner, and *Issuer* and *Bank* work together to generate two sign keys (K_{SI}, K_{SC}) and one validate key (K_P). It is noted that our scheme assures that no one can know the three keys simultaneously unless collusion. The scheme is stated as follows:

Key generation:

Issuer: Generate two keys K_{SI} and K_P. K_{SI} and K_P are large primes.
Compute $PK = K_{SI} * K_P$
Send PK to Bank, K_P to Consumer via a secure channel.

Bank: Select two large prime numbers p and q such that $\gcd(PK, \Phi(n))=1$, $n=p*q$, $\Phi(n)=(p-1)*(q-1)$.
Generate key $K_{SC} = PK^{-1} \bmod \Phi(n)$.
Send K_{SC} to Consumer via a secure channel.

A coin dual signature *CS* $= (coin)^{K_{SI} K_{SC}} \bmod n$

Verification of coin dual signature: $CS^{K_P} \stackrel{?}{\equiv} coin \bmod n$

4 The Proposed Approach

The following table shows the notations used in this paper.

Notation	Description
C	Consumer
B	Bank
M	Merchant
I	Issuer
TA	Trusted Authority
N	An amount of coins withdrawn by a consumer
Life	Coin expiration time
T	Timestamp
$H^n()$	$H^n() = H(H^{n-1}())$, *H* is a hash function
PK_X, SK_X	Long-term public key pair of user *X*
PPK, *PSK*	Pseudo public key pair of consumer
$< Data > K_X$	*Data* being encrypted with key K_X
$Sig_X(Data)$	A digital signature given by user *X* $Sig_X(Data) = Data, < H(Data) > SK_X$

4.1 Register Phase

As mentioned above, each consumer must register with the Trusted Authority (*TA*) to obtain a Payment Certificate before engaging any transaction. The consumer first generates a pseudo public/private key pair (*PPK/PSK*), and then sends his real identity (*C*) and pseudo public key (*PPK*) to Trusted Authority. After Trusted Authority authenticates the consumer, it issues a Payment Certificate (*PCert*) containing the *PPK* with its signature. The *PCert* is encrypted with the public key of Issuer and sent to consumer. The consumer's real identity and his associated *PPK* are recorded in Trusted Authority's database. The register flow is as follows.

(R1) $Consumer \rightarrow TA :< Sig_{SK_C}(C, PPK) > PK_{TA}$

(R2) $TA \rightarrow Consumer :< PCert > PK_C$

$PCert =< Sig_{SK_{TA}}(TA, PPK) > PK_I$

4.2 Withdrawal Phase

When a consumer withdraws *N* coins, the following six steps (W1) ~ (W6) are executed. For the sake of simplicity, any message transmitted in this phase is assumed to have a sender's digital signature to provide the source authentication and message integrity, and encrypt the message by receiver's long-term public key.

(W1) $Consumer \rightarrow Bank : C, PCert, Sig_{PSK}(N, T)$

Consumer sends out a withdrawal request message (W1) containing his real identity (*C*), Payment Certificate (*PCert*), the amount of withdrawal coins (*N*) and timestamp (*T*). N and T are signed with consumer's pseudo private key *PSK* to be a proof of the validity of the withdrawal, but not reveal the consumer real identity while Bank requests Issuer to make coins in step (W2). After authenticating the consumer and checking his bank balance, the bank sends a mintage request message (W2) to Issuer.

(W2) $Bank \rightarrow Issuer : B, Life, PCert, Sig_{PSK}(N, T)$

The mintage request message contains *Bank*'s identity (*B*), coin expiration time (*Life*), *PCert* and $Sig_{PSK}(N, T)$ copied from the withdrawal message. Issuer gets consumer's pseudo public key *PPK* from *PCert* to verify $Sig_{PSK}(N, T)$ to ensure the withdrawal indeed requested by an anonymous consumer, not the bank. This prevents *Bank* from stealing money.

As the scheme described in Section 3, *Issuer* first selects two large primes as the keys K_{SI} and K_P for coin signing and coin verifying. Coins must be accompanied with a certificate called as *Coin Certificate* (*CCert*) to be a proof of the validity of coins. However, to prevent *Issuer* from making coins of his own accord and *Bank* from knowing the correspondence between *Coin Certificate* and the consumer, *Coin Certificate* is prepared by *Issuer* and blindly signed by *Bank*.

Issuer prepares *Coin Certificate* in the form of *Bank*'s identity (*B*), coin verification key (K_P) signed by *Issuer*, consumer's pseudo public key (*PPK*), and coin expiration time (*Life*). *Issuer* chooses a random number *r* encrypted with *Bank* public key PK_B to blind the digest of *Coin Certificate* and computes $PK=K_{SI}*K_P$. After that, *Issuer* sends a blind-signature request message (W3) to *Bank*.

(W3) $Issuer \rightarrow Bank : H(CCert)^* < r > PK_B, PK$

$CCert = B, < K_P > SK_I, PPK, Life$

$PK = K_{SI} * K_P$

In this step, *Bank* does the following tasks: (1) Select two large prime numbers *p* and *q* such that gcd(*PK*, Φ(*n*))=1, *n*=*p***q*, Φ(*n*)=(*p*-1)*(*q*-1). (2) Generate key $K_{SC}= PK^{-1}$ mod Φ(*n*). (3) Sign *H*(*CCert*)*<*r*>PK_B by key SK_B to get <*H*(*CCert*)>SK_B**r* and put it in the blind-signature response message (W4) being sent to *Issuer*.

(W4) $Bank \rightarrow Issuer :< H(CCert) > SK_B * r, n$

Issuer first takes off the blinding factor *r* from <*H*(*CCert*)>SK_B**r* and appends it to *CCert* to form $Sig_{SK_B}(CCert) = CCert, < H(CCert) > SK_B$. Then *Issuer* uses the hash chain technique [11-13] to make *N* coins. The representation of *N* coins is as: $coin_i = (c_i, c_i')$, $1 \leqq i \leqq N$, c_N is a random number generated by *Issuer*, $c_i = H(c_{i+1}), c_i' =< c_i > K_{SI}$, where H(.) is a strong one-way hash function. After that, *Issuer* encrypts *N* coins and $Sig_{SK_B}(CCert)$ with consumer pseudo public key *PPK*, which is got from Payment Certificate (*PCert*) in withdraw request message (W1), and put them in the mintage response message (W5) being sent to *Bank*.

(W5) $Issuer \rightarrow Bank :< coin_i, 1 \le i \le n, Sig_{SK_B}(CCert) > PPK$

In this step, *Bank* simply makes a signature on the coin sign key K_{SC} generated in step (W3) and relays $< coin_i, 1 \le i \le n, Sig_{SK_B}(CCert) > PPK$ of mintage response message (W5) to consumer.

(W6) $Bank \rightarrow Consumer :< coin_i, 1 \le i \le n, Sig_{SK_B}(CCert) > PPK, Sig_{SKB}(K_{SC})$

Consumer decrypts the message by using his pseudo private key *PSK* to get *N* coins: $< coin_i, 1 \le i \le n, Sig_{SK_B}(CCert) >$, and the coin sign key K_{SC}.

4.3 Payment Phase

In the payment phase, we assume that *Consumer* spends *k* coins ($coin_i$, $coin_{i+1}$, …, $coin_{i+k}$). Recall that $coin_i = (c_i, c_i'), c_i = H(c_{i+1}), c_i' =< c_i > K_{SI}$. *Consumer* must sign the first $coin_i$ with the key K_{SC} before he spends *k* coins. Then *Consumer* prepares *Payment Information* in the form of *Merchant*'s identity (*M*), the transaction timestamp (*T*), *Coin Certificate* ($Sig_{SK_B}(CCert)$), *k* coins (the first $coin_i$ has dual signature by *Issuer* and *Consumer*). The *payment information* is signed with consumer's pseudo private key (*PSK*) to enforce non-repudiation. After that, *Consumer* sends a payment message (P1) to *Merchant*.

(P1) $Consumer \rightarrow Merchant :< Order, Sig_{PSK}(PI) > PK_M$

$PI = M, T, Sig_{SK_B}(CCert), < c_i' > K_{SC}, c_{i+k}$

To verify the legitimacy of coins, *Merchant* first verifies $Sig_{SK_B}(CCert)$ to make sure that it is signed by *Bank*. He then gets *PPK* and K_P from *CCert*. He uses PPK to verify $Sig_{PSK}(PI)$ and computes $<< c_i' > K_{SC} > K_P \stackrel{?}{\equiv} H^k(c_{i+k})$ to decide accept or reject the payment.

(P2) $Merchant \rightarrow Consumer : Success / Abort$

After verifying the legitimacy of coins, *Merchant* either ships a receipt and the merchandise to consumer, or aborts the transaction if illegal coins are used. *Merchant* keeps $Sig_{PSK}(PI)$ for depositing.

4.4 Deposit Phase

In this phase, *Merchant* deposit coins by sending a deposit message to *Bank*. The deposit message contains *Merchant* identity (*M*), deposit timestamp (*T*), and payment information $Sig_{PSK}(PI)$ for each transaction.

(D1) $Merchant \rightarrow Bank : Sig_{SK_M}(M, T, Sig_{PSK}(PI))$

Upon receiving the deposit message, *Bank* must not only verify the legitimacy of coins as *Merchant* does in step (P1) but also detect if any double spending occurs. To detect double spending, Bank has a database with one entry for each used coin chain having valid *Life* time. Each entry records $Sig_{PSK}(PI)$ of the latest transaction of the coin owner.

There are two cases when *Bank* verifies the coins: (1) there is no entry for the coin chain in *Bank* database, i.e., the coin chain has never been used before. (2) *Bank* database has already an entry for the coin chain. For the former case, it is no doubt that there is no double spending, and what Bank needs to do is to create an entry for the coin chain. As for the latter one, from the database, *Bank* can know which c_{i-1} is the last coin being spent, by validating $c_{i-1} \stackrel{?}{\equiv} H^{k+1}(c_{i+k})$, no double spending can be sure if the equation holds. Otherwise *Bank* will ask *Trusted Authority* to find out who is the owner of the double spending coins.

5 Security Analysis

- **Unforgeability**: The *Coin Certificate* is signed by *Bank* and the coin verifying key K_P is signed by *Issuer*. No one can forge the coin unless he has both the private keys of *Bank* and *Issuer*. Neither *Bank* nor *Issuer* can forge coin. This is because *Bank* can't impersonate consumer to request *Issuer* to mint coins without consumer's signed withdrawal slip. It is also impossible for *Issuer* to mint coins of his own accord. This is because *Bank* will not blind sign the *Coin Certificate* if he did not send mintage request message to *Issuer*. The coins can not be used with *Coin Certificate* signed by *Bank*.
- **Unexpandability**: *Consumer* can not extend any additional valid coins from the *N* legal coins. Each $coin_i = (c_i, c_i')$. Although consumer can find out c_0 by executing $c_0 = H(c_1)$, since he does not know the K_{SI}, he can't extend another coin $c_0' = < c_0 > K_{SI}$. Also he cannot extend c_{N+1} from c_N because of the one-way property of hash function.
- **No framing**: Though *consumer*, *Issuer* and *Merchant* contact the coins, but only *consumer* can spend the coins since a legal coin must be signed by K_{SI} and K_{SC}. This is because *Issuer* only knows K_{SI}, and it is impossible for *Merchant* to generate payment information signed by consumer's pseudo private key (*PSK*).
- **Double-spending detection**: As we discuss in deposit phase, *Bank* can detect any coin being used twice and find out the real identity of the double-spending consumer with the help of *Trusted Authority*.
- **Anonymity**: In our scheme, there is no party knew the real identity of *Consumer* and the coins he owns in the same time. In withdrawal phase, although *Bank* knew *Consumer* who he is, the coins are encrypted by *Consumer's PPK*, *Bank* cannot know any correspondence between the coins and their owner. *Issuer* knows what the coins are represented, but he does not know the coins owner. In payment phase, there is no information about the real identity of *Consumer* sent to *Merchant*. In addition, the coin does not contain any information about *Consumer*. Therefore, *Merchant* does not know whom he trades with.
- **Efficiency**: The verification of *Coin Certificate* is only processes once by *Bank* when the coin chain is used for the first time. Thereafter, coin's verification and double spending detection can be quickly processed by applying hash function. For *Merchant*, the verification of the coins is only done for the first coin, not the whole coin chain.

6 Conclusion

An important property in the micropayment system is the minimal computational overhead during a transaction. Our system uses coin chain technique to make coin that the verification of coin can be done quickly by hash computation. Besides, the proposed dual signature scheme makes the verification of coins not needed to back to the first coin of the coin chain. Even though the dual signature scheme needs three keys, no extra key management is needed for *Bank* or *Issuer*.

Our scheme ensures that the coins could only be used by their owner, and protects the privacy of the consumer. However the bank traces the real identity of the consumer only when the consumer illegally uses coins. Issuer does a lot of RSA encryption while making coins. That needs powerful computing CPU. Our future work will focus on improving system efficiency by applying Elliptic Curve Cryptography.

Acknowledgement

This work is supported by National Science Council under the grant number NSC 95-2221-E-224-017-MY2.

References

[1] N. Asokan, P. A. Janson, M. Steiner, M. Waidner, "The state of the art in electronic payment systems", *IEEE Computer*, Vol.30, No.9, pp.28-35, Sept. 1997.

[2] L. de Carvalho Ferreira, R. Dahab, "A Scheme for Analyzing Electronic Payment Systems", *Proceedings of The 14th Annual IEEE Computer Security Applications Conference*, pp.137-146, 1998.

[3] T. Ebringer, P. Thorne, Y. Zheng, "Parasitic Authentication To Protect Your E-Wallet", *IEEE Computer*, Vol.33, No.10, pp.54-60, Oct. 2000.

[4] M. A. Sirbu, "Credits and debits on the Internet", IEEE Spectrum, Vol.34 , No.2, pp.23–29, Feb. 1997.

[5] S. H. Low, N.F. Maxemchuk, S. Paul, "Anonymous Credit Cards", IEEE Networking, IEEE/ACM Transactions on , Vol.4, No.6, pp.809–816, Dec. 1996.

[6] M. Jakobsson and M. Yung, "Revokable and Versatile Electronic Money", *Proceedings of the 3rd ACM conference on Computer and communications security*, pp.76-87, Jan. 1996.

[7] "Secure Electronic Transactions", in *VISA and MasterCard*, <http://www.mastercard.com/SET>, 1996

[8] M. Bellare, J.A. Garay, R. Hauser, A. Herzberg, H. Krawczyk, M. Steiner, G. Tsudik, E. V. Herreweghen, M. Waidner, "Design, Implementation, and Deployment of the iKP Secure Electronic Payment System", *IEEE Journal on Selected Areas in Communications*, Vol.18, No.4, pp.611-627 , Apr. 2000.

[9] M. Sirbu, J.D. Tygar, "NetBill an Internet commerce system optimized for network-delivered services", *IEEE Personal Communications*, Vol.2, No.4, pp.20-25, Aug. 1995.

[10] R.L. Rivest, A. Shamir, L. Ademna, "A method for obtaining digital signatures and public-key cryptosystems", *Communications of the ACM*, Vol.21, No.2, pp.120-126, Feb. 1978.

[11] E. Mohammed, A.E. Emarah, K. El-Shennawy, "A blind signature scheme based on ElGamal signature", *Proceedings of EUROCOMM 2000, Information Systems for Enhanced Public Safety and Security*, pp.51–53, 2000.

[12] R.H. Deng, Y. Han, A.B. Jeng, T. Ngair, "A new on-line cash check scheme", *Proceedings of the 4th ACM conference on Computer and communications security*, pp.111-116, Apr. 1997.

[13] D. Chaum, "Online cash checks", *Proceedings of the Advances in Cryptology Proceedings of Eurocrypt'89*, Lecture Notes in Computer Science 434, Springer-Verlag, pp.288-293, Sept. 1989.

[14] K. Q. Nguyen, Y. Mu, V. Varadharajan, "Secure and efficient digital coins", *Proceedings of the 13th Annual IEEE Computer Security Applications Conference*, pp.9-15, 1997.

[15] K. Q. Nguyen, Y. Mu, V. Varadharajan, "Micro-digital money for electronic commerce", *Proceedings of the 13th Annual IEEE Computer Security Applications Conference*, pp.2-8, 1997.

[16] L. Buttyan, S. N. Ben, "A Payment Scheme for Broadcast Multimedia Streams", *Proceedings Sixth IEEE Symposium on IEEE Computers and Communications*, pp.668-673, 2001.

[17] Z. Yang, W. Lang, Y. Tan, "A New Fair Micropayment System Based on Hash Chain", *Proceedings of the 2004 International Conference on e-Technology, e-Commerce and e-Service*, pp.139-145, 2004.

[18] L.C. Wuu, C.M. Lin, W.F. Wang, "Anonymous and Transferable Coins in Pay-Fair Ecommerce", *IEICE Transaction on Information and Systems*, Vol.E89-D, No.12, pp.2950-2956, 2006.

[19] V. Varadharajan, K. Q. Nguyen, Y. Mu, "On the design of efficient RSA-based off-line electronic cash schemes", *Theoretical Computer Science 226*, pp.173-184, 1999.

[20] H. Wang and Y. Zhang, "Untraceable off-line electronic cash flow in e-commerce", *Proceedings of the 24th IEEE Computer Science Conference*, pp.191-198, 2001.