An Anonymous Postpaid Micropayment Scheme

Shin-Jia Hwang* and Han-Hsin Pang
Department of Computer Science and Information Engineering,
TamKang University, Tamsui, Taipei Hsien, 251, Taiwan, R.O.C.
E-mail: *sjhwang@mail.tku.edu.tw

Abstract—*Anonymity is a necessary property for postpaid micropayment schemes to protect the customers' privacy. Customers also gain the shopping convenience in postpaid way. In 2006, Huang first proposed an anonymous postpaid micropayment scheme that needs a trusted bank to provide customers' anonymity and finance services. However, the building and maintenance of trusted banks causes the impracticality of Huang's scheme. To remove trusted banks, a postpaid micropayment scheme with revocable customers' anonymity is proposed. Our scheme not only overcomes the impractical frauds in Huang's scheme but also is more efficient and practical than Huang's scheme. Moreover, our scheme is the first micropayment scheme adopting concurrent signature schemes to provide customer anonymity.*

Keywords: *Micropayment, anonymity, concurrent signatures, digital signatures, payment, smart cards*

1. INTRODUCTION

A micropayment scheme is an electronic payment scheme with low cost for small value transactions. Small value transactions occur when customers want to buy some products/services with low price. Transactions are small value, so the profit of each transaction is also small. The cost for small value transactions should be low, too. However, the proposed macropayment schemes [1, 3, 4, 8, 13, 16, 17, 18, 20] have high computation and communication costs, so these macropayment schemes are not suitable for small value transactions. To design payment with low cost, many researchers have developed some micropayment schemes [2, 9, 10, 11, 15, 19, 21].

In order to reduce the transaction cost, customers' privacy is not first considered to design micropayment schemes. However, the customers' privacy is still an important issue for micropayment schemes. To protect customers' privacy, customers' anonymity is considered to design micropayment schemes. Some researchers started proposing anonymous micropayment schemes [14, 23].

Due to [22], two kinds of anonymity are introduced. One is payment anonymity and the other is payer untraceability. Payment anonymity means that it is impossible to obtain any information about customers' identity from each single transaction. Therefore, payment anonymity protects customers' anonymity requirement during transactions. To protect customers' anonymity, this kind of anonymity is not enough. Anonymous payment transactions may be clustered into many groups such that the transactions in the same group are from the same customer. Then, a malicious attacker observes these transactions in the same group to disclose the customer's real identity. To be compared with payment anonymity, payer untraceabiltiy means that malicious attackers cannot trace customers' identities by collecting and analyzing lots of anonymous transactions, from the same customer. Therefore, the anonymity provided by payer untraceability is stronger than the one provided by payment anonymity.

Moreover, postpaid micropayment schemes are more attractive to customers than prepaid micropayment schemes. In the prepaid scheme, customers have to give their money before buying goods/servers while, in postpaid scheme, customers buy many goods/servers before paying these transactions. Customers are more interested in using postpaid micropayment schemes because they can pay after purchasing goods/services. In this situation, postpaid micropayment schemes offer attractive methods for customers.

The design of anonymous prepaid micropayment schemes is easier than the one of anonymous postpaid micropayment schemes [14]. The postpaid scheme must need a trace mechanism to deal with customers' debits. Moreover, the postpaid scheme is not risk-free. As mentioned in Yen's scheme [24], the risk of postpaid schemes is that a customer has inadequate credit to pay for the purchased goods/services. If the postpaid scheme provides customers the anonymity, then a trace mechanism for customers' debts must exist. However, the trace mechanism is the opposite of the customers' anonymity. Therefore, this opposite is the challenge to design the anonymous micropayment schemes.

Huang [12] first proposed a postpaid micropayment scheme with revocable customers' anonymity in 2006. Huang's scheme satisfies payer untraceability and provides a revocable customers' anonymity. Customers' anonymity cannot be traced because the anonymous

transactions cannot be clustered. Moreover, once disputes occur among customers, merchants, and the bank, an anonymity revocation mechanism is provided to solve disputes.

However, Huang's scheme has to involve a trusted bank that provides not only trustworthy customers' identity protection but also trust financial services. The building and maintenance of the trusted bank is a hard and heavy task. Once the trusted bank's database is compromised by malicious attackers, not only the customers' identities will be disclosed, but also the customers will lose their money stored in the trusted bank. So the implementation of Huang's scheme is hard and impractical.

Without loss of the anonymity and the postpaid property, an anonymous postpaid micropayment system is proposed to remove the trusted bank. The absence of trusted banks causes that the new scheme is more practical than Huang's scheme. Moreover, our scheme also overcomes the possible fraud in Huang's scheme.

Besides payment anonymity and payer untraceability, our postpaid scheme satisfies the following properties: Double spending prevention, anonymity revocation, and independence of anonymity provider and financial institution.

The following section is the review of concurrent signature schemes because the new scheme is proposed based on the concept of concurrent signature schemes. Section 3 describes the model, notations and assumptions in our scheme. Our new scheme is proposed in Section 4 and the security analysis of our scheme is given in Section 5. The security properties and performance comparison between Huang's scheme and ours is given in Section 6. The final section is our conclusions.

2. REVIEW OF CONCURRENT SIGNATURE SCHEMES

The concurrent signature scheme is briefly reviewed here. The formal definition of a concurrent signature scheme is first described. Then, the concurrent signature protocol is reviewed.

2.1 Concurrent signature algorithm

A formal definition of a concurrent signature scheme is first given. A concurrent signature scheme is comprised of the following algorithms: SETUP, ASIGN, AVERIFY, and VERIFY.

SETUP: A probabilistic algorithm that on input a security parameter *l*, outputs descriptions of the set of participants U, the message space M, the signature space S, the keystone space K, the keystone fix space F, and a function KGEN: K $\rightarrow$F. The algorithm also outputs the public keys PK_i of all the participants, each participant retaining their private key SK_i, and any additional system parameters p.

ASIGN: A probabilistic algorithm on the input (PK_i, PK_j, SK_i, e, m) outputs an ambiguous signature ASig= (s, e) on m, where $e \in F$, PK_i and $PK_j \neq PK_i$ are distinct public keys, SK_i is the private key corresponding to PK_i, and a message $m \in M$, $s \in S$, $e \in F$.

AVERIFY: An algorithm taking the input S=(ASig, PK_i, PK_j, m) outputs accept or reject, where ASig= (s, e), $s \in S$, $e \in F$, PK_i and PK_j are public keys, and $m \in M$.

VERIFY: An algorithm takes an input (k, *S*), where $k \in K$ is a keystone, *S*= (ASig, PK_i, PK_j, m), ASig= (s, e) with $s \in S$, $e \in F$, PK_i and PK_j are two distinct public keys, and $m \in M$. The algorithm first checks if KGEN(k)=e. If KGEN(k)$\neq$e, it terminates with output reject; otherwise it output the result of AVERIFY(*S*).

The signature ASig is called an ambiguous signature and any pair (k, ASig) is called a concurrent signature, where k is a valid keystone for ASig.

2.2 Concurrent signature protocol

The concurrent signature protocol between an initial signer A and a matching signer B is described below. Suppose that A's public and private keys are PK_A and SK_A, and B's public and private keys are PK_B and SK_B after executing SETUP.

Step 1: A picks a random keystone $k \in K$, and computes e=KGEN(k). For the message $m_A \in M$, A computes the ambiguous signature $ASig_A$= ASIGN(PK_A, PK_B, SK_A, e, m_A) and sends ($ASig_A$, e) to B.

Step 2: After receiving ($ASig_A$ e), B verifies ($ASig_A$ e) by performing AVERIFY($ASig_A$, PK_A, PK_B, M_A). If AVERIFY($ASig_A$, PK_A, PK_B, M_A)$\neq$accept, B aborts. Otherwise B generates ambiguous signature $ASig_B$= ASIGN(PK_B, PK_A, SK_B, e, m_B) for the message $m_B \in M$. B sends $ASig_B$ back to A. Note that B uses the same value e to generate $ASig_B$.

Step 4: After receiving $ASig_B$, A performs AVERIFY($ASig_B$, PK_B, PK_A, m_B) to validate

$ASig_B$. If AVERIFY($ASig_B$, PK_B, PK_A, m_B)= accept, A sends keystone k to B.

Note that inputs (k, $ASig_A$, PK_A, PK_B, m_A) and (k, $ASig_B$, PK_B, PK_A, m_B) will now both be accepted by VERIFY.

3. MODEL, NOTATIONS AND ASSUMPTIONS OF OUR SCHEME

3.1 Model of our scheme

The model for our scheme is consisted of six kinds of members: A smart card authority, brokers, a pseudonym authority, merchants, customers, and smart cards. The work of each member is described below.

Smart Card Authority (**SCA** for short)

The smart card authority is a trusted manufacturer which is responsible to manufacture and initialize smart cards. Then smart cards are sent to the broker.

Broker (**B** for short)

The broker accepts the account registration from customers using their pseudonyms. For each customer, the broker issues one smart card to generate payword chains. The broker also accepts the account registration from merchants with their real identities. In addition, the broker is responsible to redeem the valid paywords from merchants, and deal with the money transference among customers' accounts and merchants' accounts.

Pseudonym Authority (**PA** for short)

A trusted pseudonym authority is responsible to certify a certificate for customers' pseudonyms. The pseudonym authority also holds the evidence to bind each pseudonym's to the corresponding real customer. Once the dispute occurs among a broker, customers, and merchants, the broker could request the PA to disclose the customer's real identity.

Merchant (**M** for short):

Merchants offer all kinds of services or goods to the customer in exchange for the paywords.

Customer (**C** for short):

Customers use paywords to purchase services or goods from merchants.

Smart card (**SC** for short**)**:

The smart card is an agent of the broker to control and manage the debt that the customer owes to the broker.

Figure 1 shows the flowchart of our scheme. This scheme consists of eight phases: Smart card generation phase, pseudonym registration phase, account registration phase, key updating phase, commitment phase, payment phase, redemption phase, and anonymous revocation phase. The purpose of the eight phases are described in brief.

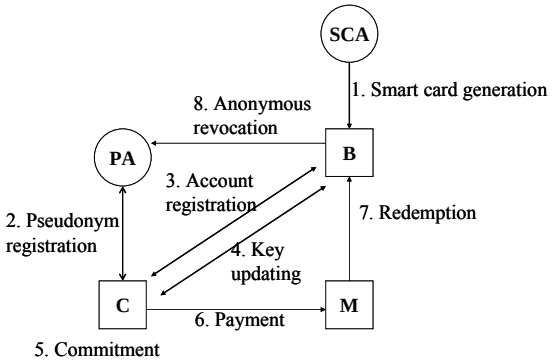

Figure 1: The flowchart of our scheme

Smart card generation phase:

The smart card authority initializes and sends smart cards to the broker. Each smart card has its own public key and secret key.

Pseudonym registration phase:

A customer must apply the PA for the certificate of his/her chosen pseudonym in advance. Then he/she uses this pseudonym to open an account in the broker. In this phase, PA and customer exchange their digital signatures with one another. The customer attempts to obtain PA's signature for his/her pseudonym as the pseudonymous certificate. PA attempts to obtain the customer's signature as the binding evidence between the pseudonym's and the real customer. To achieve fair exchange of signatures, the concurrent signature scheme is used.

Account registration phase:

A customer takes his/her pseudonym certificate to open an anonymous account in the broker. If the pseudonymous account opens successfully, the broker

sends a smart card to him/her. Similarly, a merchant also opens accounts in the broker. The major difference is that the merchant's account is not anonymous.

Key updating phase:

To achieve payer untraceability, the smart card has to create a new and unused current public key for each payword chain. Then the smart card registers this new current public key on the broker. The smart card's current public key must be unique. The broker generates a certificate for the smart card's current public key. Then, the certificate is sent back the smart card.

Commitment phase:

After obtaining the certificate from the broker, the smart card is authorized to generate a payword chain and sign a commitment for this payword chain. By using the commitment and the payword chain, the customer is able to purchase goods or services in one specific merchant.

Payment phase:

Customers use smart cards to send the merchant the smart card's current public key, the broker's certificate for the smart card's current public key, the paywords, and the commitment to the payword chain to purchase goods. The merchant could validate the received payword by verifying the certificate and the commitment. If the payword is valid, the merchant sends the goods or services to the customer and records necessary payment information.

Redemption phase:

After a period of time, the merchant redeems money from the broker for his received paywords. The broker validates the received commitment and the certificate of the smart card's current public key. If both commitment and certificate are valid, the broker continues to verify the paywords sent from the merchant. The broker transfers the payment from customer's anonymous account to the merchant's account. The broker also records necessary information of the paywords to prevent double redemption.

After completing the redemption, the customer's debt on the smart card should be also decreased. However, the broker cannot find the smart card to decrease the debt for the customer's anonymity. To decrease the debt on the smart card, the smart card has to actively inform the broker a debet-decreasing message. Then the broker will help the smart card to decrease the debt on smart card with an authorized way.

Anonymous revocation phase:

In order to resolve disputes among customers, brokers, and merchants, the pseudonym authority is responsible to revoke this customer's anonymity. To revoke the customer's anonymity, the broker first discloses the customer's pseudonym as a keystone to the pseudonym authority. Using this pseudonym, the pseudonym authority obtains a signature of the relation between the customer and the corresponding pseudonym to revoke the anonymity. To regain the anonymity, the customer registers a new pseudonym at the pseudonym authority.

3.2 Notations

The notations used in our new scheme are defined.

SCA, B, PA, M, C, SC: The notations SCA, B, PA, M, C, and SC represents the smart card authority, the broker, the pseudonym authority, the merchant, the customer, and the smart card, respectively.

ID_{SCA}, ID_B, ID_{PA}, ID_M, ID_C, ID_{SC}: The notations ID_{SCA}, ID_B, ID_{PA}, ID_M, ID_C, and ID_{SC} are the identifications of the smart card authority, the broker, the pseudonym authority, the merchant, the customer, and the smart card, respectively.

ID_P: ID_P is a corresponding pseudonym of the customer.

p, q: Two large public primes p and q satisfying $q|(p-1)$.

g: g is a generator in Z^*_P with order q.

H(): H() is a public one-way hash function.

SK_i/PK_i: SK_i and PK_i represent entity i's secret key and public key, respectively.

K_{ij}: K_{ij} denotes a session key between entity i and j.

Exp: Exp is an expired date of a payword chain given by B.

I_{SC}: I_{SC} is an upper bound of credit line which B gives to C.

n: n is the length of payword chain.

OWE: The notation OWE is a field maintained by each smart card, which records the customer's debts.

debit: The notation debit is a field maintained by the broker, which records the debts which the customer has redeemed.

$Sign_{SK_i}(m)$: $Sign_{SK_i}(m)$ is a digital signature generated by using entity i's secret key SK_i on message m

$Verify_{PK_i}(m)$: $Verify_{PK_i}(m)$ is a verification of the digital signature by using entity i's public key PK_i.

$Cert_i$: $Cert_i$ is public key PK_i's certificate.

$En_{PK_i}()$: $En_{PK_i}()$ is an asymmetric encryption function by using entity i's public key PK_i.

$E_{K_{ij}}()$: $E_{K_{ij}}()$ is a symmetric encryption function by using the session key between entity i and j.

KGEN(): KGEN() is a public one-way hash function.

ASIGN(): ASIGN() is a function used to generate an ambiguous signature.

$ASig_i$: The notation $ASig_i$ is an ambiguous signature signed by entity i.

AVERIFY(): AVERIFY() is a function used to verify the validation of an ambiguous signature.

VERIFY(): VERIFY() is a function used to verify the validation of a concurrent siganture.

3.3 Assumptions

Some assumptions in the new scheme are described here. Assume that PA's, SCA's, C's, B's, and M's public key PK_i are certified by the trusted certificate authority. Due to authentication, two secure channels must exist. One is between the pseudonym authority and the customer, the other is between the customer and the broker. Our scheme assumes that PA, B, SCA are trustworthy. PA should be trustworthy because PA is able to know the bind between a customer C and his/her pseudonym ID_P in B or PA. B should be trustworthy in finance since B is responsible to the money transfer among customer and merchants. SCA is also trustworthy since SCA will choose an asymmetric key pair (SK_i, PK_i) uniquely for each SC**.**

4. OUR NEW SCHEME

In this section, the details of each phase are described, respectively.

4.1 Smart card generation phase

In this phase, SCA prepares and initializes a lot of smart cards in advance. Firstly, SCA chooses unique ID_{SC} for each smart card. Then, SCA randomly chooses the secret key of the smart card $SK_{SC} \in Z^*_q$, and computes the public key of the smart card $PK_{SC} = g^{SK_{SC}} \bmod p$, where $Z^*_q = \{1, 2, \ldots, q-1\}$. Finally, SCA sets OWE= 0. Then, SCA generates a signature $Cert_{SC} = Sign_{SK_{SCA}}(ID_{SC}, PK_{SC})$ as a certificate of SC with SCA's secret key SK_{SCA}. SCA stores $\{ID_{SC}, SK_{SC}, PK_{SC}, Cert_{SC}, OWE\}$ on SC and stores $\{ID_{SC}, PK_{SC}, Cert_{SC}\}$ in SCA's database. Finally, SCA sends these SCs to the broker B.

After receiving SCs, B verifies the certificate $Cert_{SC} = Sign_{SK_{SCA}}(ID_{SC}, PK_{SC})$ stored on SC by SCA's public key PK_{SCA}. If the verification is valid, B randomly chooses a session key K_{SB} and stores K_{SB} in SC.

4.2 Pseudonym registration phase

In this phase, a customer C attempts to obtain a certificate of a pseudonym ID_P signed by the pseudonym authority PA. On the other hand, PA wants to obtain C's signature as an evidence of the relation between the pseudonym ID_P and Customer C. To achieve fair exchange of signatures, the concurrent signature scheme is used. The scheme is started by the customer C. PA can not get the customer C's integral signature right away until he/she releases the keystone ID_P, i.e. he/she opens an account on broker B using his/her pseudonym ID_P. To get an account on the broker B, Customer C has to release ID_P. When the broker B gives PA to revoke the cusomter's anonymity by showing the pseudonym ID_P, fair exchange of signatures is achieved. The details are described below:

Step 1: The Customer C generates an ambiguous signature and sends it to PA.

Step 1-1: Choose a random pseudonym ID_P as keystone and $SK_P \in Z^*_q$.

Step 1-2: Compute $e = KGEN(ID_P)$ and $PK_P = g^{SK_P} \bmod p$. Here, KGEN() is an one-way hash function. C generates a key pair (SK_P, PK_P) for this pseudonym.

Step 1-3: C generates an ambiguous signature $ASig_C = ASIGN(PK_C, PK_{PA}, SK_C, e, m_C||ID_C||PK_P)$ and sends $\{ASig_C, m_C||ID_C||PK_P, e\}$ to PA, where m_C contains some specification for the certification of identity PK_P.

Step 2: PA verifies C's ambiguous signature and generates and sends another one to C.

Step 2-1: Run $AVERIFY(ASig_C, PK_C, PK_{PA}, m_C||ID_C||PK_P)$ to verify whether or not the ambiguous signature $ASig_C$ is signed by C. If $ASig_C$ is signed by C, PA stores $\{ID_C, e, m_C, PK_P\}$; otherwise the process stops.

Step 2-2: Generate another ambiguous signature $ASig_{PA} = ASIGN(PK_{PA}, PK_C, SK_{PA}, e, m_{PA}||PK_P)$ and send $\{ASig_{PA}, m_{PA}||PK_P\}$ back to C.

Step 3: After receiving $\{ASig_{PA}, m_{PA}||PK_P\}$, the customer C uses the keystone ID_P to perform the verifying function VERIFY(ID_P, $ASig_{PA}$, PK_{PA}, PK_C, $m_{PA}||PK_P$). If the verifying function returns true, the customer C can get a concurrent signature as Certificate of PK_P signed by PA ($Cert_P$= {ID_P**,** $ASig_{PA}$}).

After C releases keystone ID_P, PA could verify whether or not the verifying function VERIFY**(**ID_P, $ASig_C$, PK_C, PK_{PA}, $m_C||ID_C||PK_P$) returns true. If VERIFY**(**ID_P, $ASig_C$, PK_C, PK_{PA}, $m_C||ID_C||PK_P$) returns true, PA will get a concurrent signature (ID_P**,** $ASig_C$) as an evidence signed by C. To protect customers' anonymity, the customer does not give PA the keystone/pseudonym. However, customers have to open anonymous accounts by using their pseudonyms. PA will obtain the customer's pseudonym ID_P when brokers show customers' pseudonyms to revoke customers' anonymity.

4.3 Account registration phase

After obtaining the certificate for PK_P and ID_P, the customer C wants to open an anonymous account in the broker B through a secure channel. The following steps show the procedure to open anonymous account.

Step 1: The customer C sends $\{Cert_P=\{ID_P, ASig_{PA}\}, m_{PA}||PK_P\}$ to the broker B.

Step 2: After receiving $\{Cert_P=\{ID_P, ASig_{PA}\}, m_{PA}||PK_P\}$, the broker B verifies $Cert_P$ by PA's public key PK_{PA}. If the $Cert_P$ is valid, the broker B assigns a smart card SC to the customer C and stores $\{ID_P, PK_P, Cert_P, ID_{SC}, PK_{SC}, Cert_{SC}, K_{SB}$, debit=0$\}$ in B's database, where debit is used to accumulate the debit.

Step 3: B sends the smart card SC to the customer C.

The merchant M has to open an account in the broker B by using his real identity ID_M. Then the merchant M shares a session key K_{BM} with the broker B.

4.4 Key updating phase

To achieve payer untraceability, the smart card SC must change and apply a new current public key PK_{SC}' for each new payword chain. After a new current public key PK_{SC}' is determined, the broker B will generate a signature on the new current public key PK_{SC}' with B's secret key SK_B. The following protocol is used for key updating.

4.4.1 Key updating protocol:

Step 1: The customer C starts the current key update request in this step.

Step 1-1: Randomly choose $x' \in Z^*_q$ and compute $g^{x'}$ mod p to generate a new current key pair (SK_{SC}' , PK_{SC}') of SC.

Step 1-2: Generate a signature $Sign_{SK_P}(H(g^{x'}))$.

Step 1-3: Send $\{Sign_{SK_P}(H(g^{x'})), x'\}$ to SC.

Step 2: The smart card SC computes $g^{x'}$ mod p and sends $\{ID_{SC}, E_{K_{SB}}(ID_{SC}, g^{x'}$ mod p, $Sign_{SK_P}(H(g^{x'}))\}$ to B. To reduce the computation cost of the smart card, SC does not verify $Sign_{SK_P}(H(g^{x'}))$.

Step 3: The broker B processes the current key updating request by the following sub-steps.

Step 3-1: Find a session key K_{SB}, the anonymous customer's public key PK_P and the smart card's public key PK_{SC} in its database, according to SC's identity ID_{SC}.

Step 3-2: Decrypt the ciphertext $E_{K_{SB}}(ID_{SC}, g^{x'}$ mod p, $Sign_{SK_P}(H(g^{x'})))$ with the key K_{SB}.

Step 3-3: Check whether or not the ID_{SC} in the ciphertext is equal to the ID_{SC} in plaintext. If they are not equal, reject this request.

Step 3-4: Verify the signature $Sign_{SK_P}(H(g^{x'}))$ with PK_P. If $Sign_{SK_P}(H(g^{x'}))$ is invalid, reject this request.

Step 3-5: Compute $PK_{SC}"= PK_{SC} \times g^{x'}$ mod p and checks whether or not $PK_{SC}"$ is unused in B's database. If $PK_{SC}"$ is used, B informs the customer C to reapply the request.

Step 3-6: Sign $Sign_{SK_B}(PK_{SC}", Exp, I_{SC})$ and send $\{Sign_{SK_B}(PK_{SC}", Exp, I_{SC}), Exp, I_{SC})\}$ back to the smart card SC. Here, the credit line I_{SC} might be changed over time. Therefore, every time the broker B signs for SC's new current public $PK_{SC}"$, B could change an appropriate I_{SC} for the customer C.

Step 3-7: Store the record $\{PK_{SC}", ID_{SC}, Sign_{SK_P}(H(g^{x'})), Sign_{SK_B}(PK_{SC}", Exp, I_{SC})\}$ in its database.

Step 4: The smart card SC updates its current key in this step.

Step 4-1: Compute $SK_{SC}'= SK_{SC}+ x'$ mod q and $PK_{SC}'= PK_{SC} \times g^{x'}$ mod p.

Step 4-2: Confirm whether or not PK_{SC}' and PK_{SC}'' are the same by validating $Sign_{SK_B}(PK_{SC}'', Exp, I_{SC})$ on the message (PK_{SC}', Exp, I_{SC}) with B's public key PK_B. If the verification is valid, it means that $PK_{SC}'= PK_{SC}''$. From now on, only PK_{SC}' is used as SC's current public key.

Step 4-3: Store $\{SK_{SC}', PK_{SC}', Sign_{SK_B}(PK_{SC}', Exp, I_{SC}), Exp, I_{SC}\}$ in its database.

4.5 Commitment phase

The broker B's signature $Sign_{SK_B}(PK_{SC}', Exp, I_{SC})$ implies that B authorizes SC to generate a new payword chain by the following protocol. Suppose that the customer wants to generate a payword chain with n paywords.

Payword chain generation protocol

Step 1: The customer C randomly chooses W_n and sends $\{n, W_n, ID_M\}$ to SC.

Step 2: SC checks whether or not OWE+ $n \leq I_{SC}$. If OWE+ $n \leq I_{SC}$, SC computes OWE= OWE+ n.

Step 3: SC generates the payword chain $(W_0, W_1, W_2, \ldots, W_n)$ and CMT= $Sign_{SK_{SC}'}(W_0, ID_M, ID_B, n)$, where $W_{i-1} = h(W_i)$, i= n, n-1, n-2, …, 1.

Step 4: SC informs C the successful message.

Now, the customer C purchases goods or service from M with the help of the smart card SC..

4.6 Payment phase

In this phase, the customer C uses his/her paywords to purchase goods or services from the merchant M. This phase consists of two protocols: **Payment setup protocol** and **further payment protocol**. These two protocols are described below.

4.6.1 Payment setup protocol

Step 1: SC sends $\{W_0, n, ID_B, Sign_{SK_B}(PK_{SC}', Exp, I_{SC}), PK_{SC}', Exp, I_{SC}, CMT\}$ to M.

Step 2: M checks whether or not the date is not over the expiry date Exp. Then, M verifies B's signature $Sign_{SK_B}(PK_{SC}', Exp, I_{SC})$ by B's public key PK_B and uses SC's public key PK_{SC}' to verify the commitment CMT. If all verifications are valid, M replies to SC that C can start shopping at M.

Step 3: M stores $\{Sign_{SK_B}(PK_{SC}', Exp, I_{SC}), PK_{SC}', Exp, I_{SC}, CMT, W_0, W_{index}, index= 0\}$.

Suppose that the current payword value W_{index} and index in the merchant's database are w_i and i, respectively. The customer C wants to buy some goods or services worth L paywords. The further payment protocol is described below.

4.6.2 Further payment protocol

Step 1: SC sends $\{PK_{SC}', W_{i+L}, i+L\}$ to M.

Step 2: M retrieves $\{Sign_{SK_B}(PK_{SC}', Exp, I_{SC}), PK_{SC}', Exp, I_{SC}, CMT, W_0, W_{index}= W_i, index= i\}$ and computes L=(i+L)-index=(i+L)-i .

Step 3: M performs the two verifications i+ L≤ n and $H^L(W_{i+L})= W_{index} = W_i$. If both two verifications are correct, M sends the goods or service to C.

Step 4: M replaces W_{index} with W_{i+L} and sets index= i+ L

Step 5: M stores $\{Sign_{SK_B}(PK_{SC}', Exp, I_{SC}), PK_{SC}', Exp, I_{SC}, CMT, W_0, W_{index}= W_{i+L}, index= i+L\}$.

4.7 Redemption phase

The merchant M uses the received paywords to redeem money from the broker B. This phase consists of three protocols: **Redemption setup protocol, further redemption protocol**, and **recovery protocol**. The three protocols are described below.

4.7.1 Redemption setup protocol

Step 1: The merchant M sends $\{ID_M, E_{K_{BM}}(Sign_{SK_B}(PK_{SC}', Exp, I_{SC}), PK_{SC}', Exp, I_{SC}, CMT, W_0, ID_M, n)\}$ to the broker B.

Step 2: B finds K_{BM} in B's database according to ID_M.

Step 3: B decrypts $E_{K_{BM}}(Sign_{SK_B}(PK_{SC}', Exp, I_{SC}), PK_{SC}', Exp, I_{SC}, CMT, W_0, ID_M, n)$ by K_{BM}.

Step 4: B checks the validation of Exp, and validates B's signature $Sign_{SK_B}(PK_{SC}', Exp, I_{SC})$ and commitment CMT. If any verification is invalid, B will reject this request.

Step 5: B stores $\{PK_{SC}', CMT, ID_M, W_0, W_{index} = W_0, index= 0, n\}$ in its database.

Suppose that the current payword and index in B's database are w_i and i, respectively. Now the merchant M wants to redeem L paywords.

4.7.2 Further redemption protocol

Step 1: The merchant M sends $\{PK_{SC}', W_{i+L}, L\}$ to the broker B.

Step 2: B retrieves the record $\{PK_{SC}', CMT, ID_M, W_0, W_{index}= W_i, index= i, n\}$ according to PK_{SC}'.

Step 3: B checks whether or not $i+L \leq n$. If $i+ L > n$, B rejects this redemption.

Step 4: B checks whether or not $H^L(W_{i+L})= W_i$. If $H^L(W_{i+L}) \neq W_i$, B rejects this redemption.

Step 5: B finds ID_{SC} by retrieving the record $\{PK_{SC}', ID_{SC}, Sign_{SK_P}(H(g^{x'})), Sign_{SK_B}(PK_{SC}', Exp, I_{SC})\}$ according to PK_{SC}'. Then B retrieves $\{ID_P, PK_P, Cert_P, ID_{SC}, PK_{SC}, Cert_{SC}, K_{SB}, debit\}$ according to ID_{SC}.

Step 6: B transfers the amount L of redemption from ID_P's account to M's account.

Step 7: B replaces W_{index} with W_{i+L} and sets index = i+ L. Then B stores the record $\{PK_{SC}', CMT, ID_M, W_0, W_{index}= W_{i+L}, index= i+L, n\}$

Step 8: B sets debit= debit+ L.

4.7.3 Recovery protocol

The recovery protocol is started by C or SC. Because C is anonymous and maybe off-line, it is hard for the broker B to contact C or SC immediately after the merchant's redemption. However, C or SC is able to contact the broker B on anytime by sending a clear message CLEAR_Request. After receiving the CLEAR_Request, B sends $\{ID_B, E_{BS}(ID_B, CLEAR\_Response, debit, Timestamp)\}$ to SC and sets debit = 0, temporarily. Then SC decrypts the message $E_{K_{SB}}(ID_B, CLEAR\_Response, debit, Timestamp)$ by the session key K_{SB}. Then SC verifies the validation of Timestamp and confirms the contents of CLEAR_Response. If both checks are correct, SC computes OWE = OWE– debit. Finally SC returns B an acknowledgement encrypted with the key K_{SB} to terminate this process.

4.8 Anonymous revocation phase

If a customer C is involved in a dispute with the broker B or the merchant M, the broker B would request the pseudonym authority PA to revoke the C's anonymity.

Step 1: B signs $Sign_{SK_B}(ID_P, PK_P)$.

Step 2: B sends $\{Sign_{SK_B}(ID_P, PK_P), ID_P, PK_P\}$ to PA.

Step 3: After receiving $\{Sign_{SK_B}(ID_P, PK_P), ID_P, PK_P\}$, PA first verifies whether or not the signature $Sign_{SK_B}(ID_P, PK_P)$ is valid.

Step 4: According to PK_P and ID_P, PA verifies whether or not Verify**(**ID_P, $ASig_C$, PK_C, PK_{PA}, $m_C||ID_C||PK_P$) is valid. If the verification is valid, the concurrent signature for $m_C||ID_C||PK_P$ is shown as $\{ID_P, ASig_C\}$.

Step 5: PA sends $\{ID_P, ASig_C, PK_C, m_C, ID_C\}$ to B.

Step 6: According to $\{ID_P, ASig_C, PK_C, m_C, ID_C\}$, B gets an evidence of the relation of the customer C and ID_P by verifying whether or not Verify(ID_P, $ASig_C$, PK_C, PK_{PA}, $m_C||ID_C||PK_P$) is valid.

When the customer C's anonymity is revoked, C's identity is found. To recovery anonymity, C has to register a new pseudonym ID_P', a new public key PK_P', and a new certificate $Cert_P'= \{ID_P', ASig_{PA}'\}$ from PA. Then the customer sends $\{Cert_P' = \{ID_P', ASig_{PA}'\}, m_{PA}'||PK_P'\}$ to obtain a new anonymous in the broker. The customer C also obtains a new smart card from B.

For easy understanding and clear explanation of our new scheme, all involved entities' records in their own databases are listed in Table 1.

Table 1.The records in each entity's databases

Entity	Records
SCA	$\{ID_{SC}, PK_{SC}, Cert_{SC}\}$
B	$\{ID_P, PK_P, Cert_P, ID_{SC}, PK_{SC}, Cert_{SC}, K_{SB}, debit=0\}$ $\{PK_{SC}', ID_{SC}, Sign_{SK_P}(H(g^{x'})), Sign_{SK_B}(PK_{SC}', Exp, I_{SC})\}$ $\{PK_{SC}', CMT, ID_M, W_0, W_{index}= W_0, index= 0, n\}$
PA	$\{ID_C, e, m_C, PK_P\}$
SC	$\{ID_{SC}, SK_{SC}, PK_{SC}, Cert_{SC}, OWE\}$
M	$\{PK_{SC}', CMT, W_0, W_{index}=W_0, index= 0\}$

5. SECURITY ANALYSIS

The security analysis and anonymity issues of our scheme are given here.

5.1 Double spending prevention

Consider the double spending problem first. The paywords cannot be spent on more than one merchants

because the commitments of payword chains are merchant-specific. Moreover, a malicious customer cannot send the same paywords to the same merchant M twice. The reason is that the merchant M records the latest payword w_i and latest index i in his/her local database at the end of each payment. So, M can detect whether or not the received paywords are used. So our scheme can prevent the double spending attack.

5.2 Unforgeability

Suppose that a malicious customer C' wants to forge paywords without the broker's endorsement and the smart card's help. C' cannot succeed because he/she does not know the private keys of both the broker and the smart card. Due to the computationally infeasible property of the discrete logarithm problem (DLP), C' cannot derive smart card's current private keys SK'_{SC} from its current public key PK'_{SC} and broker's private key SK_B from the broker's public key PK_B. Therefore, both private keys are secure. Moreover, assume that the smart card is tamper-resistant without leaking its private key SK'_{SC}.

5.3 Non-repudiation

A customer C cannot deny the paywords he/she used. Each payword chain has a commitment which is signed by using the smart card's current secret key SK'_{SC}, where $SK'_{SC}= SK_{SC}+ x' \bmod q$. To provide evidences, the customer C provides his/her signature $Sign_{SK_P}(H(g^{x_1}))$ by the pseudonymous secret key SK_P. Now the anonymous customer with pseudonymous public key PK_P cannot deny the paywords he/she used. Although the signature $Sign_{SK_P}(H(g^{x_1}))$ by using pseudonymous public key PK_P, another evidences is required to show the relationship between PK_P and ID_C. The relationship between PK_P and ID_C is guaranteed by the concurrent signature $\{ID_P, ASig_C\}$. Due to the non-repudiation of the concurrent signatures, the customer cannot deny the paywords he/she used.

5.4 Anonymity

Assume that PA is trustworthy and SC is tamper-resistant without leaking customer's identity. Being based on these assumptions, the anonymity of our scheme is discussed from brokers', merchants' and anyone's viewpoints, respectively.

Broker B's viewpoint

Our scheme provides customers only payment anonymity from brokers' viewpoint. Broker B only knows the pseudonym ID_P of a customer, so B does not know the real identity ID_C without the help of the PA. But the broker B is able to classify transactions according to pseudonyms. Therefore, our scheme provides customer payment anonymity but not payer untraceabilty. In order to increase customers' privacy, each customer may own more than one pseudonyms and open many anonymous accounts on the same broker B.

Merchant M's viewpoint

In our scheme, M cannot learn C's identity ID_C or trace C's payments. In our scheme, the commitment CMT of each payword chain is verified by different public key PK'_{SC}. These CMTs do not contain any information about customers, so CMTs disclose nothing about customers. Moreover, the merchant M cannot classify CMTs because each CMT is verified by different and random public key PK'_{SC}. However, PK'_{SC} is randomized, so M cannot use it to trace customers. So, payer untraceability is achieved from M's viewpoint.

Anyone's viewpoint

From anyone's viewpoint, the best situation is that he/she receives the same information as the one of the merchants. Our scheme provides customers payer untraceability from merchants' viewpoints, so our scheme also provides custoemrs' payer untraceability from anyone's viewpoints.

5.5 Anonymity Revocation

When disputes occur among the customers, the merchants, and the broker, the broker could request PA to revoke the customers' anonymity. Therefore, a customer C can not deny the pseudonym ID_P he/she registered on PA after anonymous revocation phase. After PA receives ID_P as the keystone from the broker, PA can get the concurrent signature $\{ID_P, ASig_C\}$ signed by the real customer C's secret key SK_C. C's real identity is disclosed by verifying whether $Verify(ID_P, ASig_C, PK_C, PK_{PA}, m_C||ID_C||PK_P)$ returns true. According to the validation of the concurrent signature $\{ID_P, ASig_C\}$, B knows the relationship between the pseudonym ID_P and the corresponding real customer C. Therefore, the concurrent signature is a relationship evidence between ID_P and the real identity ID_C.

6. COMPARISONS

6.1 Comparison of security properties

The comparison between our new scheme and Huang's scheme is given. The common security properties for micropayment schemes are described below. The first property is double spending prevention that each one electronic coin is spent only one time. The payment anonymity property means that no one can determine the

payer's identity according to a single payment except the trusted third party PA. The payer untraceability means that customer's payments cannot be traced, by discovering the payer's identity based on two or more payments of the same payer. The anonymity revocation property means that there exists a revoking mechanism, being performed by a trusted party, to find the identities of anonymous customers. A postpaid micropayment means that the customer can pay the cash after receiving their goods/services. The independence between the trusted party and the broker means that there is no relationship between the trusted party and the broker.

Table 2 summarizes security properties comparison between our scheme and Huang's scheme. In Huang's scheme, a Trust Bank (TB for short) is needed to provide anonymity and financial transference services. The TB plays not only the trusted party protecting customers' identities but also a financial institution. If there is some fraud occurring in TB, the damage is serious about not only anonymity but also finance. To reduce the damage caused by TB's fraud, the two distinct trusted parties are needed to provide anonymity and financial transference, respectively and independently. In our scheme, a trusted pseudonym authority only provides anonymity and anonymity revocation services. The financial transference service is provided by the broker with anonymous accounts. So our scheme not only supplies the properties of Huang's scheme, but also divides the trusted third party and the broker definitely. Therefore, our scheme overcomes the possible fraud in Huang's scheme. However, in broker's view, our scheme provides payment anonymity while Huang's scheme provides payer untraceability. But our scheme is more efficient than Huang's scheme due to the performance analysis.

Table 2:Security property comparison

Properties \ Schemes		Our scheme	Huang's scheme
Double spending prevention		✓	✓
Payment anonymity	Brokers	✓	✓
	Merchants	✓	✓
	Anyone	✓	✓
Payer untraceability	Brokers	✗	✓
	Merchants	✓	✓
	Anyone	✓	✓
Anonymity revocation		✓	✓
Postpaid		✓	✓
Independence of anonymity provider and financial institution		✓	✗

6.2 Performance analysis

The computational performance is measured by six kinds of cryptographic functions: The signature generation and verification, the symmetric encryption and decryption, the public-key encryption and decryption, the hash function, the blind signature generation, and the concurrent signature generation and verification. To generate concurrent signatures, the two signers have to generate ambiguous signatures first. So the generation cost of an ambiguous signature is also the same as the generation cost of a concurrent signature. In the proposed concurrent signature schemes [**7**], the verification cost of concurrent signatures is equal to the sum of the verification cost of ambiguous signatures and the verification cost of keystones. Because the verification cost of keystones is much less than the verification cost of ambiguous signatures. In the following, the verification cost of a concurrent signature is used to estimate the verification cost of ambiguous signatures.

Some notations to represent computational cost are defined below. The notation Sign denotes the generation cost of one signature while the notation Ver denotes the verification cost of one signature. The notation EN denotes the cost of one symmetric encryption operation and the notation DE denotes the cost of one symmetric decryption operation. The notation Hash denotes the cost to execute the hash function once. The notation PE is used to denote the cost of one public key encryption operation. The notation PD is used to denote the cost of one public key decryption operation. The notation blind denotes the cost of executing a blinding function. The notation unblind denotes the cost of executing an unblinding function. The notation Sig-C denotes the generation cost of one concurrent signature. The notation Ver-C denotes the verification cost of one concurrent signature.

Because Huang's scheme is also an anonymous postpaid micropayment scheme, the performance comparison between Huang's scheme and our scheme is given here. To achieve anonymity, our scheme needs an additional phase: pseudonym registration phase. Moreover, instead of using blind signature schemes in Huang's scheme, our scheme adopts concurrent signature

schemes. These phases can be classified into two classes. One class consists of the repeated phases. The another class consists of the phases that occur seldom. The repeated phases are key updating phase, commitment phase, payment phase, and redemption phases. The seldom phases are smart card generation phase, pseudonym registration phase, account registration phase, and anonymous revocation phase.

The computational cost for each repeated phase is discussed first because these phases occur frequently. To construct one payword chain, key updating phase and commitment phase should be executed. In the key updating phase, the cost paid by our scheme is almost the same to Huang's scheme. The only difference is that our scheme needs one additional Sign by the broker B and one additional Ver on SC. However, in the commitment phase, the additional computational cost in Huang's scheme is more than the cost in our scheme. In Huang's scheme, the customer C first applies for a certificate of SC's current public key from the bank B. Then, SC could generate the payword chain anonymously. To achieve this purpose, the additional cost paid by B is one EN, one DE and one Sign. The customer C's additional cost is one blind. SC's additional cost is one EN, one DE, one unblind, and one Ver. Totally, to generate one payword chain, the computational cost paid by our scheme is less than the cost in Huang's scheme.

In the payment phase, the computational cost paid of our scheme and Huang's scheme is the same. In the redemption phase, the computational cost paid by our scheme is less than the cost paid by Huang's scheme. The reason is that, in Huang's scheme, one more transformation is performed. In order to redeem money from the customer's account in the TB, the bank B sends the commitment and the paywords to TB. Therefore, in Huang's scheme, TB has to pay the extra computation cost, one DE, two Vers, n Hashes, and one EN. Therefore, our scheme is more efficient than Huang's scheme during these phases that frequently occur.

In the following, the computational cost for each seldom phase is discussed. In the smart card generation phase, the process of each smart card in our scheme is similar to the one in Huang's scheme. The computational cost is almost the same. However, the generation of each smart card is performed once; the computational cost comparison is ignored.

In the pseudonym registration phase of our scheme, the customer C performs additional one hash, one Sig-C, and one Ver-C. So the customer needs 1 Hash+1Sig-C+1 Ver-C. On the others hand, the PA needs one Sig-C and one Ver-C. However, in our scheme, any customer can use the same pseudonym for many payword chains. This cost is reduced by sharing with many payword chains.

Now consider the cost paid in account registration phase. In our scheme, the customer directly applies his/her certificate for PK_P and ID_P to open an anonymous account in the broker through secure channels. Only the broker needs one Ver-C to validate the certificate that is also a concurrence signature in our scheme. However, in Huang' scheme, the customer has to first open an account in TB through secure channels. The customer personally obtains one smart card SC and the certificate of PK_{SC} from TB. Then the smart card helps the customer obtain the session key between the customer and the bank. To obtain the session key, the computational cost paid by the smart cards is one Sign and one PD. On the other hand, the bank needs two Vers and one PE. Therefore, in account registration phase, our scheme is more efficient than Huang's scheme.

In the anonymous revocation phase, when a customer is involved in a dispute, the merchant M or the bank B only needs to inform TB to revoke the customer's anonymity in Huang's scheme. However, to achieve this purpose, our scheme needs to execute additional one Sign and one Ver on B and one Ver-C on PA. The computational cost paid in the anonymous revocation phase of our scheme could be shared by the case without disputes occurred. To recover the anonymous, Huang' scheme needs no computational cost but our scheme needs the computational cost to execute pseudonym registration phase again. Although our scheme needs additional computational cost for anonymity revocation/recovery, it is worthy to remove the need of a trusted bank. Table 3 demonstrates the performance comparison between Huang's scheme and our scheme in detail.

Table 3: Computational performance comparison

	Huang's scheme	Our scheme
Smart Card Generation		
Pseudonym Registration		**C**: 1 Hash + 1 Sig-C + 1 Ver-C **PA**: 1 Ver-C + 1 Sig-C
Account Registration	**B**: 2 Vers + 1 PE **SC**: 1Sign + 1 PD	**B**: 1 Ver-C

Key Updating	**TB**: 1 DE + 1 Ver **C**: 1 Hash + 1 Sign **SC**: 1 EN	**C**: 1 Hash + 1 Sign **B**: 1 DE + 1 Sign + 1 Ver **SC**: 1 EN + 1 Ver
Commitment	**B**: 1 DE +1 EN + 1 Sign **C**: 1 blind **SC**: 1 EN + 1 DE + n Hashes+ 1 Sign + 1 unblind+1 Ver	**SC**: 1 Sign + n Hashes
Payment	**M**: 2 Vers + n Hashes	**M**: 2 Vers + n Hashes
Redemption	**TB**: 1 DE+ 2 Vers+ n Hashes+ 1 EN **B**: 1 DE+ 2 Vers + n Hashes + 1 EN **M**: 1 EN **SC**: 1 DE	**B**: 1 DE + 2 Vers + n Hashes + 1 EN **SC**: 1 DE M: 1 EN
Anonymous Revocation		**B**: 1 Sign + 1 Ver **PA**: 1 Ver-C

7. CONCLUSIONS

A new anonymous postpaid micro-payment system is proposed. Our scheme is more practical than Huang's scheme because of the deletion of trusted banks in Huang's scheme. Then the impractical frauds in Huang' scheme caused by the existence of trusted banks are removed in our scheme. The transaction performance between customers and merchants in our scheme is more efficient than the one in Huang's scheme. The redemption efficiency between merchants and brokers in our scheme is better then the one in Huang's scheme. The transaction between customers and merchants and the redemption occur frequently, these costs dominant the performance of one micropayment scheme, so our scheme is more efficient than Huang's scheme. Our scheme could attract more people using the micropayment scheme in electronic commerce.

Most micropayment schemes provide anonymity with the help of blind signature schemes [5, 6]. Instead of blind signature schemes, our scheme is the first scheme adopting concurrent signature schemes [7] to provide the anonymity for a micropayment scheme.

REFERENCES

[1] Anand, R. Sai and Madhavan, C. E. Veni, "An Online, Transferable E-Cash Payment System," *Advance in Cryptology – INDOCRYPT 2000,* LNCS, Vol. 1977, New York: Springer-Verlag, pp. 93–103, 2000.

[2] Bellare, M., Garay, J., Hauser, R., Herzberg, A., Krawczyk, H., Steiner, M., Tsudik, G. and Waidner, M., "iKP – A Family of Secure Electronic Payment Protocols," *Proceeding of 1st USENIX workshop on Electronic Commerce*, pp. 89-106, 1995.

[3] Brands, Stefan, "Untraceable Off-line Cash in Wallet with Observers," *Advances in Cryptology – CRYPTO '93*, LNCS, Vol. 773, New York: Springer-Verlag, pp. 302–318, 1993.

[4] Chan, Agnes, Frankel, Yair, and Tsiounis, Yiannis, "Easy Come – Easy Go Divisible Cash," *Advances in Cryptology – EUROCRYPT '98*, LNCS, Vol. 1403, New York: Springer-Verlag, pp. 561–575, 1998.

[5] Chaum, D., "Blind signatures for untraceable payments," *Advances in Cryptography-Proceeding of Crypto '82*, New York: Springer-Verlag, pp. 199- 203, 1983.

[6] Chaum, D., Fiat, A., and Naor, M. "Untraceable Electronic Cash," *Advances in Crytology – CRYPTO'88*, LNCS, Vol. 403, New York: Springer-Verlag, pp. 21–25, 1988.

[7] Chen, L, Caroline, Kudla and Kenneth, G. Paterson. "Concurrent Signatures," *Advances in Cryptology - EUROCRYPT 2004*, LNCS, Vol. 3027, *Berlin:* Springer-Verlag, pp. 287–305, 2004.

[8] Frankel, Yair, Tsiounis, Yiannis, and Yung, Moti, "Indirect Discourse Proofs: "Achieving Efficient Fair Off-Line E-Cash System," *Advance in Cryptology – ASIACRYPT '96*, LNCS, Vol. 1163, New York: Springer-Verlag, pp. 286–300, 1996.

[9] Furche, A. and Wrightson, G., "SubScrip- An Efficient Protocol for Pay-Per-View Payments on the Interent," *Proc. 5th International Conference on Computer Communications and Networks (ICCCN '96)*, Rockville, MD, Oct. 16-19, 1996.

[10] Glassman, S., Manasse, M. S., Abadi, M., Gauthier, P., and Sobalvarro, P., "The Millicent Protocol for Inexpensive Electronic Commerce," *World Wide Web Journal, Proceeding of 4th International World Wide Web Conference*, Boston, MA: O'Reilly, 1996, pp. 603–618.

[11] Herberg, Amir, "Micropayment," in *Payment Technologies for E-Commerce*, Kou, Weidong Ed., New York: Springer, 1998, pp. 245-282.

[12] Huang, C.-W., "A Postpaid Micropayment Scheme with Revocable Customers' Anonymity," Master Thesis, Tamkang University, Taiwan, R.O.C, 2006.

[13] Jakobossn, Markus and Yung, Moti, "Revokable and Versatile Electronic Money," *Proceeding of the 3rd ACM Conference on Computer and Communications Security*, India: ACM press, pp.79–87, 1996.

[14] Lin, S.-Y., "*Design and Cryptanalysis of Micropayment Schemes*," Master Thesis, National Central University, Taiwan, R.O.C, 2004.

[15] Manasee, M. S., "The Millicent Protocols for Electronic Commerce," *Proceeding of 1st USENIX workshop on Electronic Commerce*, pp. 117–123, 1995.

[16] MasterCard and VISA "Secure Electronic Transactions," http://www.setco.org/ set.html

[17] Mu, Yi, Nguyen, Khanh Quoc, and Varadharajan, Vijay, "A Fair Electronic Cash Scheme," *Topics in Electronic Commerce: Second International Symposium – ISEC 2001*, LNCS, Vol. 2040, New York: Springer-Verlag, pp.20–32, 2001

[18] Neuman, B. C. and Medvinsky, G., "NetCheque, NetCash and the Characteristic of Internet Payment Services," *Proceeding of MIT Workshop on Internet Economics 1995*, 1995

[19] Rivest, R. L. and Shamir, A., "PayWord and MicroMint: Two Simple Micropayment Schemes," *Proceeding of Security Protocols Workshop*, LNCS 1189, New York: Springer-Verlag, 1997, pp. 69-87.

[20] Sirbu, M. and Tyger, T. J., "NetBill: An Electronic Commerce System Optimized for Network Delivered Information and Services," *Proceeding of IEEE CompCon '95*, San Francisco, CA, USA, March 1995, pp. 20-25.

[21] Stern, J. and Vaudenay, S., "SVP: a Flexible Micropayment Scheme," *Proceeding of* Financial *Cryptography*, LNCS, Vol. 1318, New York: Springer-Verlag, pp. 161-172, 1997.

[22] Tsiaris, T. and Sthephanides, G., "The Concept of Security and Trust in Electronic Payments," *Computers & Security*, Vol. 24, Issue 1, pp. 10-15, 2005.

[23] Tsou, J.-H., "The Study of Electronic Payment Scheme," Master Thesis, Tamkang University, Taiwan, R.O.C, 2005.

[24] Yen, S.-M., "PayFair: A prepaid internet micropayment scheme ensuring customer fairness," *Computers and Digital Techniques, IEE Proceedings*, Vol. 148, Issue 6, November 2001, pp. 207–213.