Available online at www.sciencedirect.com

Procedia Technology 6 (2012) 930 – 938

International Conference on Communication, Computing and Security [ICCCS-2012]

On-line Electronic Payment System using signcryption

Arpita Mazumdar[a] and Debasis Giri[b]

[a]*Department of Master of Computer Applications*
St. Mary's Technical Campus Kolkata, Kolkata-700 126, India

[b]*Department of Computer Science and Engineering*
Haldia Institute of Technology, Haldia-721657, India

Emails: arpita.smtck@gmail.com and debasis giri@hotmail.com

Abstract

In this paper, we present an on-line electronic payment system for low-value transaction. We use the concept of signcryption for each communication between any pair of entities and offers token updation instantly as an add-on feature to existing on-line e-cash systems where token is the payment instrument (containing electronic cash) which acts as an electronic wallet. In our system, the token is issued and authenticated by Issuing Bank (token issuer). Merchant verifies the user and sends authenticated transaction details to Issuing Bank. Later Issuing Bank verifies approval of both customer and merchant. The merchant is also verified by its financial institution. Unlike the existing e-payment system question of double spending of e-cash arises because each transaction are made uniquely identifiable and updation is instantly done by Issuing bank. Hence no separate protocols are needed to be developed for handling disputes. The proposed scheme provide anonymity, authenticity, confidentiality and fairness.

Keywords: Signcryption; Security; On-line; E-commerce; E-cash;

1. Introduction

The growth of the Internet, in the last years, has created an electronic market place for goods and services. Trading tangible goods is not the only way to earn money in this new virtual market place. Information or more generally intangible goods will become important gradually. While at this moment, many sites offer intangible goods free and this may change soon. Producers will sell knowledge, executable programs, images, music or videos over the Internet and deliver these goods electronically. Electronic payment system plays a crucial role, acts as a backbone of this virtual market place. For security consideration, cryptographic primitives are used in electronic payment system. If a payment system is to succeed on the Internet, the computational efforts in using these primitives need to be optimized. Many authors proposed schemes (Zheng 1997), (Anand & Madhavan 2000), (Steve Glassman & Sobalvarro 1995),

* Corresponding author. Debasis Giri; fax: +03224-252800 .
E-mail address: dgiri@hithaldia.in

2212-0173
doi:10.1016/j.protcy.2012.10.113

(He & TC-Wu 1999), (Hwang & Sung 2006), (Erik-Oliver Blass & Strufe 2009) for electronic payment systems.

In this work, we consider electronic payment systems in which the payment instrument is token (which is an electronic wallet) containing electronic cash (E-cash). Our proposed scheme is based on on-line e-payment systems where the transfer of electronic money between the payer (customer) and payee (merchant) takes place in the presence of a third party (usually, a Bank), that guarantees the authenticity of the token being used in a transaction. Verification of both customer and merchant are done by their respective financial institute respectively. Token information and transaction details are kept confidential. Both authentication and confidentiality walk on hand-in-hand simultaneously by signcryption technique [1]. To our best knowledge, none of the existing electronic payment system follows signcryption technique. The proposed scheme is so designed that it guarantees fairness to both the parties, namely the customer and the merchant. By fairness, we mean that none of the parties can dishonour a transaction. As each transaction is made unique and token containing e-cash value is updated instantly by Issuing bank, therefore double spending does not arise. The token has a certain lifetime. After each transaction, token need not be renewed. The balance amount can be carried over in the next transaction.

2. Brief overview of Anand and Madhavan's protocol

In this section, we describe Anand and Madhavan's e-cash payment protocol (Anand & Madhavan 2000) which guarantees anonymity, fairness and transferability. There are three parties involved in the basic coin exchange mechanism: the Payer, designated as the Customer (C), the Payee, called a Merchant (M) and a Verifying Authority (VA). In a transaction the coins are transferred from C to M and the coins are verified by the VA. The VA's job is two-fold: First, he has to verify that the coin has not been spent previously and next, he needs to affix his signature along with requisite information on the coin to allow the merchant M to spend the coin later.
There are three basic protocols namely, 1) Coin Withdrawal/Deposit protocol; 2) Basic Coin Exchange protocol; and 3) Resolution Protocol.

We use the following notations for the rest of the paper.

- S_X(MESG): A signature on a message, MESG, by an entity X
- H(.): A strong collision resistant one-way hash function.
- f(.): A strong collision resistant one-way hash function
- E_k(.): A symmetric-key encryption algorithm characterized by key K
- D_k(.): A decryption algorithm characterized by key K
- K_{XY}: A symmetric key between two entities X and Y

In this system, a coin (COIN), is a bit string consisting of three parts. The format of the COIN is shown below.
$[S_B(\text{SNO, DENM, EXPD, TS})]$, $[S_{VA_0}(VA_1, \text{TS}_0, \text{H(SNO, TS)})]$, $[VA_0]$ The first part has the fields, serial number (SNO), denomination of the coin (DENM), expiry date of the coin (EXPD) and the timestamp of issue (TS). This part is signed by the bank using its private key. The second part consists of the name of the next verifying authority (VA_1), a timestamp (TS_0) at which verification is been done and the hash of serial number (SNO) and timestamp of issue (TS) which are fields of the first part. The second part is then signed by the present verifying authority (VA_0). At issue stage, the present verifying authority is the bank B itself. The third part of the coin is the name of the present verifying authority (VA_0). The first part of the coin remains unchanged throughout the lifetime of the coin, that is till its expiry date.

The basic Coin Exchange Protocol of this scheme is depicted in Figure 1 below.

Coin verification : The verification of the COIN is done as follows: First, VA_i verifies from the second part that he is indeed the current valid verifying authority. Next, VA_i checks if the SNO appearing on the COIN is listed in his database. If not, then the COIN is authentic and VA_i proceeds to sign the COIN. If the COIN is listed in his database, he checks the timestamp appearing on the second part of the COIN. If this timestamp is greater than the time stamp in the database corresponding to the SNO of the COIN, then the COIN is authentic. If both of the above conditions are violated then the COIN is treated as double spent and a REJECT signal is conveyed to M. The entries that need to be made in the verifying authority's database when a COIN is found to be authentic are: (a) the SNO, if it is not already

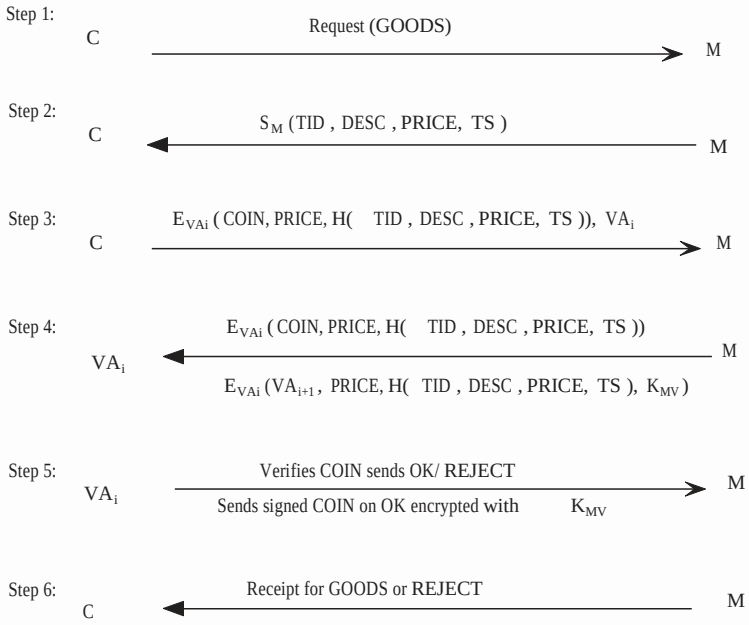

Fig. 1. Basic Coin Exchange Protocol

in the database and (b) the timestamp (TS) of previous verification which appears in the second part. If the COIN is verified as being authentic, the verifying authority, VA_i, needs to sign it. This signing is done by replacing the second part of the COIN. The second part will now contain the new timestamp (TS_i) at which the COIN is being verified, the next verifying authority's name (VA_{i+1}) and the hash value that existed in the replaced part. Of course, he may also verify that the hash value is correct by doing the hash computation. The third part of the COIN is replaced by the name of the present verifying authority, VA_i.

- Coin being verified by VA_i:
  S_B(SNO, DENM, EXPD, TS), $S_{VA_{i-1}}$ (VA_i, TS_{i-1}, H(SNO, TS)), VA_{i-1}
- Coin after Signature by VA_i:
  S_B(SNO,DENM,EXPD,TS),S_{VA_i}(VA_{i+1},TS_i,H(SNO,TS)),VA_i
- Coin Withdrawal Protocol:
  The diagram below explains Coin Withdrawal Protocol

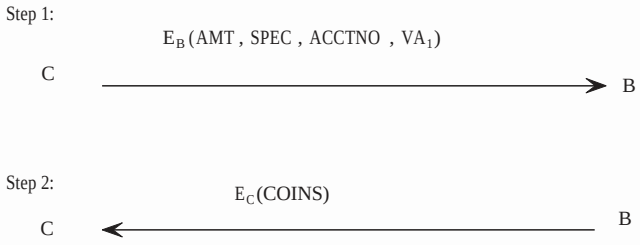

Fig. 2. Coin Withdrawal Protocol

[AMT: amount; SPEC: the change required; ACCTNO: account number]

- Coin Deposit Protocol:
  The diagram below explains Coin Deposit Protocol

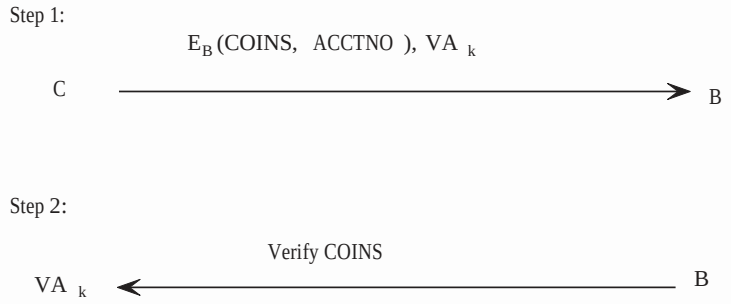

Fig. 3. Coin Deposit Protocol

Resolution Protocol
Step 1: C sends a resolve request with the signed message obtained from merchant M and a hash of the coins he had used in the transaction to the verifying authority VA_i at which the coin was sent.
Step 2: VA_i on receipt of the request checks the signature of M and if found valid checks when the coins spent were authenticated by it. If the claim of C is found correct, VA_i directs M to transfer the goods. If M does not accept then the coins transferred to it are invalidated by sending an appropriate message to VA_{i+1}. Also, the coins spent by C are restored to it so that C does not lose money in the aborted transaction.

3. Proposed scheme

3.1. Overview

The proposed E-cash system consists of following four entities.

1. Issuing Bank (IB) who issues token, validates existing tokens and exchanges E-cash(token) for real money.
2. Customer (C) who can buy tokens (updatable) from IB with exchange of real money and can purchase intangible products from Merchants, paying via electronic media.
3. Merchants (M) who can accept tokens in payment for information items.
4. Acquiring Bank (AB) where Merchants have accounts. AB validates the merchant.

In a transaction, the following events are taken place:
At first, Setup is constructed by a central authority (third party).
Step 1 (C ⇒ IB): The customer pays IB for the token.
Step 2 (IB ⇒C): IB issues signed token for C and send securely to C.
Step 3 (C ⇒M): C inserts token, choose an item from M's homepage and sends order information (OI) to M. C also sends some secret information for IB (via M) which helps it to verify C.
Step 4 (M ⇒IB): Merchant appends price details, and its own id (ID_M) to OI. Then M forwards the signcrypted modified OI to IB.
Step 5 (IB ⇒AB): IB retrieves the token information with its private key. It verifies the customer and ensure that merchant forwarded token message is genuinely sent by Customer and the OI is the corresponding requisition detail requested by the Customer in this transaction. IB also verifies its own signature and then sends merchant details to AB for confirmation.
Step 6 (AB ⇒IB): AB verifies and sends acknowledgement to IB.
Step 7 (IB ⇒M): IB sends ok/reject signal to merchant.
Step 8 (M ⇒C): Merchant sends receipt for sold items/reject to customer.
Step 9 (IB ⇒AB): IB updates e-cash value, transaction id, transaction time in the token and then transfers the fund and sends transaction details to AB.
Step 10 (AB ⇒IB): AB updates M' s account and sends confirmation to IB.

Step 11 (IB ⇒C): On receiving the confirmation IB commits the transaction and further sign the modified token and send acknowledgement to C.
Step 12 (AB ⇒M): AB sends acknowledge to merchant.
Note: X ⇒Y means X sends some information to Y.
In our system, the format of the token information (TI) is shown below

TOKENID	SEQNO.	CUSTID	EXPIRES	TS	VALUE	PROPS

TOKENID : Unique identification number of a token; SEQNO. : Unique sequence number for each transaction; CUSTID : Customer identification number; EXPIRES : Lifespan of a token; TS : Time stamp of a transaction; VALUE : Monetary value; PROPS : other information;

Moreover the format of Order Information (OI) is shown below

TOKENID	ITEMCODE	SEQNO.

ITEMCODE : unique code for an item for sale.

The block diagram of the proposed scheme is shown in the following Figure 4.

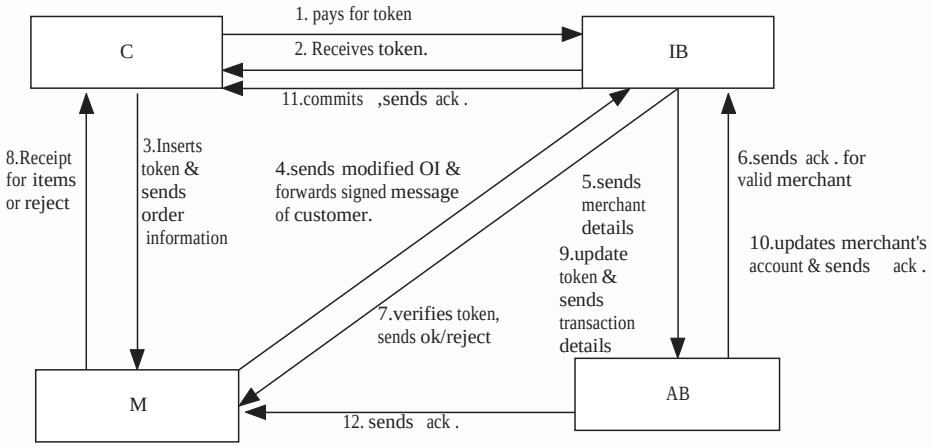

Fig. 4. Block diagram of proposed scheme

3.2. Detailed description of the Scheme

In the following, we describe the setup phase of the proposed scheme.
Setup There exists a central authority who generates the system wide public parameter p (large prime) and g (generator over Z_p^*). Also another prime q is chosen such that $q|p-1$. Let U be an user in the system. Then $x_u \in Z_q^*$ is private key and $y_u = g^{x_u} \bmod p$ is the corresponding public key of the user U. Hence (x_B, y_B), (x_u, y_u) and (x_m, y_m) are the key pairs of Bank, Customer and Merchant respectively.

Steps in the figure 4 are elaborately described below.
Step 1: The Customer pays IB for a token.
Step 2 (Generation of signcrypted token information): IB sends token to the customer using signcryption. Signcrypted message is generated by the IB using its private key and C's public key, C can verify the signcrypted message and recover the original message with the help of C's private key and IB's public key.
Signcrypted token information (TI) as (c, h, s, v) is generated by IB for C.

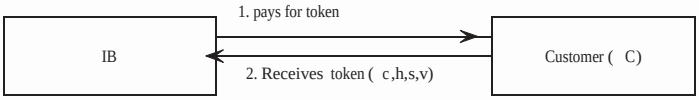

Fig. 5. Token issuance

1. Compute encryption key K as $K = f(TI)^{x_B} \bmod p$, where f is a one-way function that maps from $\{0,1\}^*$ to Z_q^*
2. Compute $c = E_K(TI)$, where E is a symmetric-key encryption algorithm.
3. Compute $v = f(TI)$
4. choose randomly $r \in Z_p^*$
5. Compute $e = f(y_u^r \bmod p, c)$
6. Compute $h = K \cdot e \bmod q$
7. Compute $s = r(h + x_B)^{-1} \bmod q$

Verification of Signcrypted message: After receiving (c, h, s, v) from IB, C verifies it as follows.

1. Compute $e = f((y_B \cdot g^h)^{s.x_u} \bmod p, c)$
2. Compute $K = h \cdot e^{-1} \bmod q$
3. Compute $TI = D_K(c)$, where D is a symmetric-key decryption algorithm.
4. Compute $v_c = f(TI)$ C verifies whether $v_c = v$. If true, C accepts the token as valid else rejects.

Step 3 (Generation of dual signature): Customer further signs the signed token ($TI \| h \| s$) and sends the dual signed token along with *OI* securely to Merchant. Merchant forwards the signed token to IB. IB can only retrieve the token value and verify its signature on it.

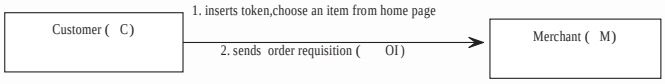

Fig. 6. placing order requisition

Signature generation by customer: Suppose, $msg = (TI \| h \| s)$. Customer signs on *msg*. Customer sends $X_2 = (c_2, h_2, s_2, v_2)$ to IB via Merchant, and sends $X_1 = (c_1, h_1, s_1, v_1)$ to Merchant, where the component of X_1 and X_2 are described below.

1. C computes encryption key $K_1 = f(OI)^{x_u} \bmod p$
2. Compute $c_1 = E(K_1, OI)$
3. Compute $v_1 = f(OI)$
4. Randomly choose $t \in Z_p^*$
5. Compute e_1=f$(y_m^t \bmod p, c_1)$
6. Compute $h_1 = K_1.e_1 \bmod q$
7. Compute $s_1 = t(h_1 + x_u)^{-1} \bmod q$
8. Randomly choose $x \in Z_p^*$
9. Compute key $k = y_B^x \bmod p$
10. Compute $c_2 = k \cdot msg \bmod p$
11. Compute $v_2 = f(msg)$
12. Randomly choose $w \in Z_p^*$

13. Compute e_2=f($y_B^w \ mod \ p, c_2$)
14. Compute $h_2 = k \cdot e_2 \bmod q$
15. Compute $s_2 = w(h_2 + x_u)^{-1} \bmod q$

Verification by Merchant: After receiving the message X_1 and X_2 from customer, Merchant verifies that *OI* has come from customer in the following manner. M cannot calculate e_2, in tern cannot decrypt c_2, it forwards (c_2, h_2, s_2, v_2) to IB.

1. Calculate $e_1 = f((y_u \cdot g^{h_1})^{s_1 \cdot x_m} \bmod p, c_1)$
2. Compute $K_1 = h_1 \cdot e_1^{-1} \bmod q$
3. Compute $OI = D(K_1, c_1)$
4. Compute $v_m = f(OI)$

Merchant verifies whether computed value v_m and value of v_1 sent by customer is same or not. If same, it ensures that *OI* has not been altered and come from an authentic customer.

Step 4 (Forwarding token information): After verification M appends its own ID that is ID_M, *price* with *OI*. This Modified Order Information, $MOI = (OI \| ID_M \| price \| ID_{AB})$ where ID_{AB} is the identity of acquiring bank. After that Merchant sends modified transaction details $X_3 = (c_3, h_3, s_3, v_3)$ to IB and also forwards signed message $X_2 = (c_2, h_2, s_2, v_2)$ of customer. Description of each parameter of X_3 are given below.

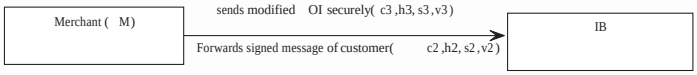

Fig. 7. Forward Token information

1. Compute encryption key $K_3 = f(MOI)^{x_m} \bmod p$
2. Compute $c_3 = E(K_3, MOI)$
3. Compute $v_3 = f(MOI)$
4. Randomly choose $d \in Z_p^*$
5. Compute $e_3 = f(y_B^d \bmod p, c_3)$
6. Compute $h_3 = K_3 \cdot e_3 \bmod q$
7. Compute $s_3 = d(h_3 + x_m)^{-1} \bmod q$

Verification: After receiving X_3, IB executes the following steps.

Verifying merchant details :

1. Compute $g^d = g^{s_3 \cdot (h_3 + x_m)} \bmod p$
2. Compute $e_3 = f((g^d)^{x_B}, c_3)$ [$= f(g^{x_B \cdot s_3 (h_3 + x_m) \bmod q} \bmod p, c_3) = f((g^{h_3} \cdot g^{x_m})^{s_3 \cdot x_B} \bmod p, c_3) = f((g^{h_3} \cdot y_m)^{s_3 \cdot x_B} \bmod p, c_3)$]
3. Compute $K_3 = h_3 \cdot e_3^{-1} \bmod q$
4. Compute $MOI = D(K_3, c_3)$
5. Compute $v_{BM} = f(MOI)$

IB verifies that computed *MOI* has come from Merchant by checking the condition $v_3 = v_{BM}$.

After receiving the message $X_2 = (c_2, h_2, s_2, v_2)$ of Customer forwarded by Merchant, IB verifies the authenticity of Customer and then retrieves the token message in the following manner.

IB recalculates the key for verifying the customer details

1. Compute $e_2 = f((g^{h_2} \cdot y_u)^{s_2 \cdot x_B} \bmod p, c_2)$

2. Compute $key = h_2 \cdot e_2^{-1} \bmod q$
3. Compute $msg = c_2 \cdot key^{-1} \bmod p$
4. Compute $v_{BC} = f(msg)$

IB checks both the conditions ($v_{BC} = v_2$) and ($TOKENID, SEQNO.$ of $TI = TOKENID, SEQNO.$ of OI) and if both are true then IB will ensure that merchant forwarded token message is genuinely sent by Customer and the OI is the corresponding requisition detail requested by the Customer in this transaction. IB also recalculates h,s from token message (TI) and matches with sent values h and s. Then IB verifies its own signature.
Step 5: IB encrypts the ($ID_M \| TS_1$) with the shared symmetric key $K_{IB,AB}$ between IB and AB and sends the encrypted message to AB for confirmation.
Step 6: If M has a valid account, AB sends $E_{k_{IB,AB}}(TS_2)$ as acknowledgement, where $TS_2 = TS_1 + 1$.
Step 7: After verifying that both customer and merchant are authentic IB sends OK signal to merchant.
Step 8: M on receipt of an OK signal sends a receipt for items to C.
Step 9: IB updates the following in the token $SEQNO.$ = next $SEQNO.$
$VALUE = VALUE - price.$
TS =current TS
IB sends $E_{K_{IB,AB}}$ ($ID_M \| price \| n_1$) to AB, where n_1 is a random nonce.
Note: Check [$TS > TS2$] must hold which means only after receiving acknowledgement (TS_2) from AB, updation is performed by IB.
Step 10: AB updates M' s account by balance= balance+price. AB confirms by sending $f(ID_{AB} \| n_1)$ to IB.
Step 11: IB commits the updation in token as well as in database. IB further signs the modified token and sends $f(SEQNO.)$ as acknowledge to customer.
Step 12: AB sends $f(ID_{AB} \| n_2), n_2$ as acknowledgement to merchant where n_2 is a random nonce.

4. Security analysis

Our proposed scheme is based on a cryptographic primitive termed 'signcryption' in which message encryption and digital signature are simultaneously performed. Therefore, it requires less computational cost and less communication overhead than conventional signature then encryption approaches.
Basically, a signcryption scheme should satisfy the following properties:
Unforgeability: It is computationally infeasible for an adaptive attacker to masquerade as the signcrypter is creating a signcrypted text.
Confidentiality: It is computationally infeasible for an adaptive attacker to find out any secret information from a signcrypted text.
Non repudiation: It is computationally feasible for a judge to settle a dispute between the signcrypter and the recipient in an event where the signcrypter denies the fact that he is the sender of the signcrypted text to the recipient.
The signcyption approach in our proposed scheme possess all the 3 properties:

- As signature and encryption is not separable, forgery attacks are not likely to occur. Calculation of e is dependent on c where c has a relation with message TI. Moreover,h is dependent on e and in tern s is dependent on h.
- As message is encrypted with unique key in each transaction, to find out any secret information from signcrypted text is computationally infeasible.
- A signcrypter can not repudiate his signature as signature is done by its private key and is related to message.

In the following, we describe the security of our scheme for step 2. Suppose, an adversary can know the value of c, h, s, v. To break the security of the scheme he has to know the value of x_B, e, r, K, message which is computationally impossible.
The adversary might solve the private key x_B from the following equation s=r(h+x_B)$^{-1}$ $mod\ q$ and given c, hv. This implies that the adversary can compute x_B from the above equation, if r is known to the adversary from $e = f(y_B^r mod\ p, c)$, which is again computationally infeasible due to the discrete logarithm problem. The security of the scheme relies on the hardness of the classical Discrete Logarithm problem in a subgroup of the group Z_p^*, for large prime p. Hence, it is

computationally infeasible to an adversary to attack in this manner. Analogously, we can prove the security for other steps of our proposed protocol.

5. Comparison

In this section, we compare our scheme with previously published schemes namely Veni Madhavan and Anand's scheme and Hwang and Sung's scheme. The comparison (in Table 1) shows that in our proposed scheme updation of Token value can be done instantly unlike the others. There is no special verification for double spending detection of token. Tokens are so designed they can not be double spent. Moreover customer's identity is not revealed by Merchant.

Table 1. Comparison of security features

Anonymity	Double spending detection	Updation of TOKEN value	Scheme	
Anonymous	Verification done by consulting database	No	Veni Madhavan and Anand's	
pseudonymous	Before each transaction C sends unique information by Encryption	No	Hwang and Sung's	
pseudonymous	Token can't be double spent	Yes	Our	

Anonymous: Customer's identity is not revealed by the Merchant.
Pseudonymous: Customer's identity is associated with TOKEN still Merchant can't reveal customer's identity.

6. Conclusion

We have designed a security protocol for on-line electronic payment system using the technique of signcryption mechanism in which token information can be updated by Issuing bank. We have then described the security analysis of our proposed scheme.

References

Anand, R. Sai & C E Veni Madhavan. 2000. Online Transferable Ecash Payment System. INDOCRYPT 2000.

Erik-Oliver Blass, Anil Kurmus, Refik Molva & Thorsten Strufe. 2009. PSP: private and secure payment with RFID. he 8th ACM workshop on Privacy in the electronic society pp. 51–60.

He, W H & TC-Wu. 1999. "Cryptanalysis and improvement of Peterson-Michels signcryption scheme." *IEE Proc.-Comput. Digit. Tech.* 146(2):123–124.

Hwang, Min-Shiang & Pei-Chen Sung. 2006. "A Study of Micro-payment Based on One-Way Hash Chain." *International Journal of Network Security* 2(2):81–90.

Steve Glassman, Mark Manasse, Martn Abadi Paul Gauthier & Patrick Sobalvarro. 1995. Millicent Protocol for inexpensive E.commerce. 4th WWW Conference Proceedings pp. 603–618.

Zheng, Yuliang. 1997. Digital Signcryption or how to achieve Cost(Signature & Encryption)<< Cost (Signature) + Cost(Encryption). 17th Annual International Cryptology Conference on Advances in Cryptology LNCS 1294, Springer-Verlag pp. 165–179.