AMVPayword: Secure and Efficient Anonymous Payword-Based Micropayment Scheme

Mona Hosseinkhani  
Department of CE and IT  
Amirkabir University of Technology  
Tehran, Iran  
e-mail: m.hosseinkhani@aut.ac.ir

Ebrahim Tarameshloo  
Department of CE and IT  
Amirkabir University of Technology  
Tehran, Iran  
e-mail: tarameshloo@aut.ac.ir

Mehdi Shajari  
Department of CE and IT  
Amirkabir University of Technology  
Tehran, Iran  
e-mail: mshajari@aut.ac.ir

Abstract—Electronic payment systems are increasingly in need of improvement as they constitute the base for electronic commerce which is rapidly dominating much trade and particularly retail. One such system that allows payments of low value is known as micropayment system the main requirement for which is naturally ‘low cost of transaction’. Payword is a micropayment system which uses hash chains and achieves better computational performance. It is a credit-based protocol involving a customer, vendor and broker. However, in this system users have to generate a new payword chain for each vendor from which they make a purchase and also their payment process is not anonymous. MVPayword was an extension to Payword which enables the user to make purchases from different vendors using only one hash chain. In this paper we shall introduce AMVPayword which is an extension to MVPayword and provides anonymity for customers. So users can be sure about the protection of their privacy in payment system.

Keywords- electronic payment; micropayment; security; anonymity; mvpayword; amvpayword

I. INTRODUCTION

Internet micropayment schemes have received growing attention recently, mostly due to the fact that these schemes exhibit the potential of being embedded in numerous internet based applications. In all of these applications, small amount of money has to be transferred during payment transactions. In addition to the computational overhead of many payment systems are not low enough for them to be incorporated in this field of e-commerce.

In 1996 the Payword system was proposed by Shamir and Rivest [2]. In this system hash functions are used to produce hash chain for payment. Since hash functions are faster than signature generation and verification [2], so Payword uses hash functions to improve its performance.

This system has two main drawbacks as following:

1) The first one is that the users must generate a different Payword chain for each vendor from which they want to make purchases. In 2009 MVPayword[1] was proposed to solve this problem and make the system more efficient.

2) The other drawback is the lack of anonymity in this protocol.

In this paper we will present an extension to MVPayword in order to protect the privacy of users in their payment transactions.

Micropayment systems are rarely anonymous. Privacy can be achieved with the use of pseudonyms [8], but when anonymity is implemented [9], the high computational cost associated is not suitable for low valued payments [10]. According to [5], a payment is anonymous if it conforms to at least one of the following criteria:

1) It preserves the privacy of a customer’s identity from a vendor so that the vendor cannot associate the customer’s identity with his purchases.

2) It preserves the privacy of a customer’s identity from a broker.

In this paper, we propose to use AMVPayword, which is an extension to MVPayword [1]. This system is an anonymous micropayment protocol which enables users to make purchases from multiple vendors using only one hash chain.

This paper is organized as follows: In section 2 we examine MVPayword. Section 3 presents AMVPayword system. Security and efficiency of our system will be analyzed in section 4. In section 5 a comparison between AMVPayword and some other similar systems will be discussed. In section 6 Future works in this area will be described and Section 7 concludes this paper.

II. MVPayword

MVPayword is a micropayment scheme proposed in 2009[1]. It enables the users to make purchases from multiple vendors using only one hash chain.

MVPayword consists of three entities: user, vendor, and broker. The user and vendor should each establish an account with the broker. The user sends his/her certificate to the vendor. Then the user creates the payword chain in reverse order by randomly picking the last payword \( w_n \), and then computing the other paywords using hash function:

\[
 w_i = h(w_{i+1}) \quad \text{for} \ i = n-1, n-2, \ldots, 0 
\]
$w_0$ is the root of hash chain while $h$ represents one hash function.

The User creates a commitment which is user-specific as well as vendor-specific and sends it to the vendor. Then the vendor sends this commitment to the broker in order to verify it. The broker examines this commitment and sends a message to the vendor containing whether it confirms the validity of that commitment or not.

If the broker accepts the validity of user’s commitment, then the payment $P$ from user $U$ to vendor $V$ will contain a payword and its index:

$$P = (w_i, i) \quad (2)$$

The vendor applies hash function $h$ to $w_i$ for $i$ times. If the result is equal to $w_j$, then the payment is verified and accepted.

At the end of validity period of commitment, the vendor sends to broker a redemption message containing user’s commitment and the last received payword from user in order to redeem paywords. Now the broker can verify received information and transfer money from the user’s to the vendor’s account.

### III. AMVPAYWORD: AN ANONYMOUS SCHEME FOR SUPPORTING MULTIPLE VENDORS USING ONLY ONE HASH CHAIN

AMVPayword is an extension to MVPayword and enables users to purchase anonymously based on MVPayword. It is quite efficient and secure. Similar to MPayword, there are three entities involved in AMVPayword: user, vendor, and broker. The user and vendor should each establish an account with a broker. The broker has a database to store information about the produced coins. There are 4 stages in AMVPayword: Receiving coins, commitment issuing and verification, payment and redemption of Paywords. These phases will be described in details.

#### A. Receiving Coin

In this phase user and broker communicate with each other. After receiving user information, the broker deduced the requested amount of money from user’s credential and gives the user a signed coin and a ticket called Insert Ticket to insert values in broker’s database.

Now the user creates a hash chain by the length of $N$ and a hash of nonce as follows:

$$H(nonce) = r_0 \quad (3)$$

The hash chain will be produced in reverse order. Each hash value represents a monetary unit that can be defined in advance. Then the user inserts some information in broker’s database with the use of his/her Insert Ticket. Fig.1 shows the messages transmitted between the user and broker in the first phase of AMVPayword.

1. $(N, T, sign_U(N, T), C_U)$

2. $(sign_B(N, T), IT) PK_U$

3. $IT, \{r_0, H(M, W_0), N\}$

**Figure 1. Receiving coin from Broker**

“$N$, “$T$”, “$sign_U(N, T)$”, “$C_U$”, “$PK_U$” indicate the requested amount of money, timestamp, user signature, user certificate and public key of the user respectively. “$IT$” represents Insert Ticket. An Insert Ticket is a once-only-usable general ticket given to the user by the broker to insert a value into the broker’s database after authenticating the user. This ticket is not user specific which means the broker can not relate the ticket to any user. “$sign_B(N, T)$” represents the user’s coin which we will call “$M$”. The broker encrypts the coin and IT with the public key of the user, thereby enabling only the user to decode it.

In order to insert values to the broker’s database, the user uses IT to insert the following triple:

$$(r_0, H(M, W_0), N) \quad (4)$$

After this insertion, only $H(M, W_0)$ and $N$ can be updated but $r_0$ should remain the same. The hash value will help vendors to prevent double spending.

#### B. Commitment issuing and verification

After the user receives his/her coin from the broker, (s)he can use it to issue commitments for each desired vendor. In AMVPayword, contents of user’s commitments determine the domain of valid paywords for related vendor. If there is any overlap between domains of different commitments, then the customer will be capable of paying some paywords to more than one vendor (double-spending). In order to make sure that paywords domains don’t overlap with each other, the vendor must check the valid domain in broker’s database. The messages transmitted for commitment issuing and verification are shown in Fig. 2

2. $r_0, H(M, H(W_i))$

3. accept/deny

**Figure 2. Commitment issuing and verification phase**

The user sends the following items to the related vendor:

- $(\text{nonce}, W_i, M, D_i) PK_B$


“Dc” , “PKB” refer to cash date and public key of broker respectively. The vendor should redeem paywords to the broker by the end of the cash date Dc.

- \( W_i, M, r_0 \)

Now the vendor calculates \( H(M, H(W_i)) \) and compares it to the hash value which is stored in the same row as \( r_0 \) in broker’s database. If these two hash values are equal in the database, then the commitment will be accepted. Otherwise, the vendor will reject this commitment to prevent double spending by user. After verification of hash value, the vendor checks broker’s database for the remaining value of the coin. If the remaining value is not sufficient, then the vendor will reject the user’s request.

C. Payment

This phase is somehow similar to that of basic Payword system. In this phase -as shown in Fig. 3, user sends new paywords to the vendor.

\[ \text{User} \quad p = (W_{i+1}, L) \quad \text{Vendor} \]

Figure 3. Payment phase

When the vendor receives a new payword from the user, s/he verifies payment by applying hash function to it (for required times). If there is not any problem, then the vendor delivers the commodities.

Now the vendor must update the stored value in broker’s database from \( (r_0, \{ H(M, H(W_i)), \text{amount} \}) \) to \( (r_0, \{ H(M, W_{i+1}), \text{amount} - L \}) \). This update must happen so that the next vendor can verify his/her correct domain value of hash chain.

D. Redemption of paywords

When all valid paywords are received by the vendor, the vendor can send to the broker the last received payword in order to redeem them. Fig. 4 shows this process.

\[ \text{Vendor} \quad (\text{nonce, } W_i, M)PKB \quad (W_{i+1}, L) \quad \text{Broker} \]

Figure 4. Payword redemption phase

After receiving \( (\text{nonce, } W_i, M)PKB \) and the last payword \( (P = W_{i+1}, L) \), the broker must do the subsequent verifications:

- The broker should use its private key to verify the encrypted part.
- The broker must check the validity of the coin. (s)he can recognize the coin since (s)he themselves have signed it.
- Apply hash function on nonce to obtain related \( r_0 \). If \( H(\text{nonce}) \) equals \( r_0 \) in broker’s database, then the broker will know that the vendor did not change the user’s commitment, otherwise it will know that the vendor is trying to cheat in protocol, and this redemption will be rejected.
- Now the broker applies hash function to “\( W_{i+1} \)” for “L” times. If the result equals \( W_i \) –which is entered in commitment–, the broker will accept the redemption process and transfer the specified amount to the vendor’s account.

Unlike other similar systems, which support multiple vendors without anonymity, in this system the user can use one commitment for several payments with anonymity.

IV. SECURITY AND EFFICIENCY OF AMVPAYWORD

In AMWPayword protocol, most of the transactions are done by applying hash functions. Because they are faster than other cryptographic operations such as digital signatures, this has improved the efficiency of the system. Moreover, in this system customers can use one hash chain for their purchases from different vendors (similar to MVPayword[1]) and this results in lower computational cost. The main advantage of this system is anonymity which ensures the protection of the user’s privacy.

Some security aspects of AMVPayword are discussed in the followings:

1) Prevention of double-spending.

In AMVPayword the vendors can examine the broker’s database and check for the valid range of paywords in commitment. If the user wants to double spend, then the verification process in the broker’s database will inform the vendor of the invalidity of the commitment. This process is done by verifying only one hash and thus it is fast and efficient. This property is one of the most important advantages of this system because vendors can independently detect double spending payments and reject them.

2) Prevention of producing and stealing paywords.

The hash chain for a specific coin in AMVPayword is produced by the owner of that coin because only the owner can get the coin and ticket from the encrypted message and store information about \( w_0 \) in broker’s database using IT. Consequently no other entity can produce or steal paywords.

3) Preventing users from spending more than credit.

The remaining value of the coins is stored in broker’s database. In effect each vendor will be able to verify the user’s credit in that coin and allow or deny the payment accordingly.

V. SIMILAR SYSTEMS AND COMPARISON BETWEEN THEM AND AMVPAYWORD
In this section, we will describe some similar payword-based systems [6],[7],[11], and then explain about the advantages of AMVPayword over these similar systems.

A. Similar systems

1) The first system.

This system was proposed by “Wang”, “Chang”, and “Lin” in 2002 [7]. In this system, when the user wants to purchase something from one vendor, (s)he creates a specific root (R_v) for that vendor and puts this root into commitment. This root is created as follows:

\[ R_v = \text{h}(w_j \oplus (U \parallel v_i)) \]  

(5)

Here \(v_i\) is the new vendor from whom the user wants to purchase, and \(w_j\) is the first unused payword in hash chain.

The user sends “\((w_{j+1}, L)\)” to the vendor for payment. Then the vendor is capable of producing \(R_v\), by applying hash function to \(w_{j+1}\) for required times. If the resulted value is equal to the one entered in the commitment, then the payment will be verified. The goal of this system is to divide the hash chain in such a way that enables the user to use that hash chain for multiple purchases.

One drawback of this system is its inability in prevention of double spending by vendors before the redemption phase on broker’s side is completed. Another shortcoming is the lack of anonymity which causes the identity of users to be revealed to broker. Because the commitments which are sent from vendor to broker in redemption phase \((M_v = \{V_v, C_v, R_v, D_v, I_{w_{j+1}}\})\) contain user’s identification.

2) The second system.

In 2003, an anonymous micropayment protocol was proposed [6]. This system was based on Payword and tried to implement anonymity in its transactions. To achieve this goal, the system used the concept of general coin. This coin is not user or vendor specific and so it is anonymous. The protocol has four steps as follows:

   a) Withdrawal subprotocol: In this step user requests the coin with the specified amount (Q). In order to avoid traceability, blind signature is used.

   b) Pre-payment subprotocol: In this step user sends the coin to the broker and requests from it to divide the coin into two coins. The value of the first coin is the maximum amount of money that the user wants to spend on a different item from the next vendor. The value of the second coin equals the remaining value in original coin.

   c) Payment subprotocol: In this phase the user sends the vendor-specific coin, \(w_i\) and index i for the vendor.

   d) Deposit subprotocol: For redemption of paywords, the vendor sends her specific coin, her identification value and \(w_i\) to broker. Now the broker can verify this information and deposit the specified value to the vendor’s account.

This protocol has some drawbacks. One of them is its high computational cost. Each time the user wants to purchase from a vendor, (s)he must send the coin to the broker and get two new coins with the broker’s signature. So broker must sign two coins and verify one signature for each vendor that the user wants to purchase from. Moreover in this protocol the user has to produce a new hash chain for each vendor and it results in a high computational cost for it. Another drawback is that when the user wants to buy from a vendor, the amount of money that (s)he wishes to spend at this vendor, must be specified in pre-payment subprotocol.

3) The third system


   a) Registration Phase: Both customer and merchant have to register with a broker. The customer shares a secret key \(K_{UB}\) with the broker, and the merchant shares a secret key \(K_{MB}\) with the broker. The customer selects a pseudonymous identity \(IDU\), which is unique to every customer. The broker selects a master secret key \(k\) and keeps it secret from others.

   b) Blinding Phase: The customer sends a redemption request to the broker before (s)he requests any service from vendor. After a 5-step transaction based on elliptic curve blinding technique, customer will receive a valid signature for his/her request.

   c) Transaction Phase: Before asking service from vendor, the customer sends a transaction request to the broker. After authenticating the customer by \(ID_{CU}\), broker creates a one-time session key for customer and vendor. Then the customer asks for service from vendor.

   d) Redemption Phase: vendor would carry out the redemption process with the broker. The broker then extracts the payment from the customer’s account and transfers the amount to the vendor’s.

B. Comparison between AMVPayword and similar systems

The first described system [7] is multi-vendor (like MVPayword [1]) but it lacks the ability to detect double spending by the vendor and it should be detected by the broker. Conversely, AMVPayword is a multi-vendor system which is capable of detecting double spending by vendor. Moreover AMVPayword has the anonymity property which is one of the important features in e-payment systems whereas this property is absent from the first system because of the commitments which are signed by customer and sent to the broker by vendor.

The second system described [6] is not multi-vendor because users have to create a new hash chain for each vendor. In AMVPayword however, by using one hash chain multiple vendors can be supported. In AMVPayword due to the use of hash operations, the computational cost is low,
while in the second system described, computational cost is high because of the signatures on coins which should be generated and verified. Moreover in the second system, users must specify the amount of money they want to spend at each vendor before payment phase, but in AMVPayword users can determine the amount of money that they want to spend, during payment phase.

The third system [11] supports multi-vendor payments and prevents overspending. Further, the scheme provides anonymous cash transaction by a blinding phase using elliptic curve cryptography and as a result increases the overall cost of the scheme. Actually this anonymity is achieved by two features. First, using blind signature and second, employing pseudonymous identity of customer in transactions (IDc). Considering the first feature, anonymity is obtained in a way that anyone takes the valid signature, (s)he cannot link this signature to the message, so the coins are untraceable. But there is an ambiguity for anonymity in broker’s side using IDcs. In redemption phase, the broker extracts the payment from customer’s account and transfers the amount to vendor’s account according to paywords. This could reveal which customer has paid his/her coins to which vendor because this scheme occupies non debit-base redemption while AMVPayword keeps anonymity of users on the broker’s side in redemption phase.

VI. FUTURE WORK

Although, the development in the area of micropayment is increasing rapidly, it should be noted that for small amount of money we should have systems with the lowest computational costs that are possible. So researchers are trying to find solutions with high security and efficiency. But gathering all aspects of security and efficiency together in one micropayment system seems to be difficult. Therefore, research in this domain is continuing. In this protocol and other similar ones, it is possible for two entities to commit fraud on the third. So removing this vulnerability is an open problem for future research.

VII. CONCLUSION

In this paper we proposed AMVPayword which is a new Payword-based micropayment system. There exist three main entities: user, vendor, and broker. This system enables the user to make anonymous purchases from different vendors using only one hash chain. In AMVPayword, vendors calculate one hash value and compare it to the value which is stored in broker’s database and according to the obtained result from database, the new commitment is accepted or denied and therefore double spending by customer will be detected by vendor. Using general coins in AMVPayword makes it an anonymous protocol and so it can provide one of the most significant features in e-payment systems. In this system, through the use of one commitment, the user can make one or more payments to each vendor depending on the issued commitment. Verification and redemption of paywords in our new system is done by vendor and broker using user’s commitment. Efficiency and security properties of AMVPayword make it a good choice for many micropayment applications in new commercial environments such as internet.

ACKNOWLEDGMENT

This research was supported by the Pars Processing Pioneer Co. under grants PPP-G0240910.

REFERENCES