A Mobile Autonomous Agent-based Secure Payment Protocol Supporting Multiple Payments

Yan Wang
Department of Computing
Macquarie University
Sydney, NSW 2109
Australia
yanwang@ics.mq.edu.au

Vijay Varadharajan
Department of Computing
Macquarie University
Sydney, NSW 2109
Australia
vijay@ics.mq.edu.au

Abstract

In agent based e-commerce applications, it is challengeable to employ one mobile agent to complete all transactions including payments due to the security consideration. In this paper, we propose a new agent-assisted secure payment protocol, which is based on SET payment protocol and aims at enabling one dispatched consumer-agent to autonomously complete the payment on behalf of the cardholder with multiple merchants. This is realized on the basis of Signature-Share scheme, Signcryption-Share scheme, and a set of security mechanisms. On one hand, the dispatched consumer-agent is able to autonomously complete the deal and the payment on behalf of the cardholder with multiple merchants for buying multiple products. On the other hand, transaction records with merchants are protected against malicious hosts.

1. Introduction

Secure payment is an important issue in e-commerce applications. Micro-payment protocols, such as PayWord [10], are suitable for completing small payments (e.g. $10 or less) while macro-payment protocols are applied for large transactions.

Some secure payment protocols are based on SSL or S-HTTP. But they are not considered to be secure enough since the credit card information is deposited in the server, where it can be read easily by anyone with access to it [11]. SET (Secure Electronic Transaction) protocol developed by VISA and MasterCard is regarded as a better protocol [1] aiming at protecting users' credit card information with important properties, such as authentication of the participants, data integrity and confidentiality.

The introduction of autonomous agents reduces the effort required from users to conduct e-commerce transactions by automating shopping activities [4, 2, 7]. An end-consumer can specify his/her preference to the agent server which dispatches an autonomous mobile agent with an encapsulated task to the remote servers of merchants for asking offers, negotiating with merchant agents and even completing payments. The results can be sent back through a message or carried back by the agent [14]. However, it is a challengeable issue to protect both mobile agents and servers when mobile agents are roaming in the network [3]. Security is important as well when agents carry critical/confidential information (e.g. credit card information), sign contracts or make payment on behalf of the consumers since the agents and their carried sensitive data will be exposed to potentially hostile environments.

Several agent-based extensions of the SET protocol have been proposed, such as the SET/A [11], SET/A+ [15] and LITESET/A+ [9], aiming at utilizing the autonomy of a mobile payment agent while ensuring the security of payments. In [12] we have analyzed the drawbacks of these protocols and proposed LITESET/A++. The goal of LITESET/A++, which is based on SET, is to enable a mobile agent to automatically and autonomously make final transactions and payment with the "best" merchant with the best offer after having performed all kinds of tasks including asking offers, and negotiating with a set of merchants.

However, all above-mentioned agent-based payment protocols assume that the payment should be made with one merchant only (i.e. the best merchant with the best offer). But in many cases, the consumer would like to buy multiple products from multiple merchants. For SET, it can be applied individually to each merchant for multiple times if multiple payments are necessary. Nevertheless, we expect that one autonomous mobile agent can complete the payments with multiple merchants without any interaction with the cardholder. The challengeable issues are 1) each merchant is determined after negotiation; it is not known in advance; 2) each merchant chooses a payment gateway (PG) according to the credit card's brand; different merchants may choose different PGs; 3) in the above-mentioned protocols, mechanisms preventing overspending and double-spending problems are not good enough.

In this paper, we present a new agent-based secure payment protocol supporting multiple payments and adopting Signature-Share scheme and Signcryption-Share scheme [9], which is based on Signcryption public key algorithm [16]. It also adopts the transaction chain to protect the transaction records, and security mechanisms to prevent overspending and double-spending.

2. Background

In this section, to understand all protocols well, we will first review SET [1]. The description of Signcryption scheme, Signature-Share and Signcryption-share schemes can be found in [9] and [12]. Notations and symbols used in this paper are listed in Table 1.

The SET protocol [1] is composed of several kinds of transactions, ranging from registration of participants, to purchase request and payment processing. There are different roles in SET. They are cardholder (C), credit card issuer, merchant, acquirer and payment gateway (PG) [1]. PG is a device of acquirer where the merchant has an account. As requested by the PG, successful payment should be finally authorized by the issuer whereafter the issuer will pay on behalf of the cardholder and the money will be deposited to the merchant's account at the acquirer.

SET uses two distinct asymmetric key pairs for each party, one for key-exchange. The corresponding public key y_{K_A} is contained in public key certificate $C_K(A)$ of participant A. The key pair (y_{K_A}, x_{K_A}) is used for encrypting and decrypting messages. Another key pair is used for the creation and verification of signatures. The signature public key of participant A is included in the signature certificate $C_S(A)$. Figure 1 depicts the purchase request phase of SET.

In SET, the key issue is to pass the payment instruction (PI) including card number, cardholder's name and expiry date to the payment gateway (PG) determined according to the brand of the cardholder's credit card that is included in purchase request (in step 1 in Figure 1). PI is encrypted by a session symmetric key K that is included in a digital envelope $E_{PG}\{K, PI\}$ passed to PG via merchant M. Finally the payment can be completed by PG without the possibility of disclosing the PI to M. Due to the limited space, readers can refer to [1] for more details.

$(K_{y_{PG_j}}, K_{x_{PG_j}})$	a pair of temporally generated session keys ($public\ key$, $secret\ key$) for payment gateway PG_j
$C_K(A)$	key-exchange certificate of participant A
$C_S(A)$	signature certificate of participant A
$c_{\to A}$	the ciphertext that should be *passed to* participant A
$E_k\{m\}$	the ciphertext of message m encrypted by key k
$E_{PG}\{K, PI\}$	the digital envelope generated by PG (= $\{E_{yK_{PG}}\{K\}, E_K\{PI\}\}$), K is a symmetric key
g	a (random) integer in $[1, \ldots, p-1]$ with order $q\ mod\ p$ (public to all)
$H(m)$	a one-way hash function applied to message m
I_A	the unique transaction number issued by participant A
KH	a keyed one-way hash function
p	a large prime (public to all)
OI	order information
PG	payment gateway
PI	payment instruction including card number, expiry date etc
q	a large prime factor of $p-1$ (public to all)
R_j	a random number chosen from $[1, \ldots, q]$
$r_{\to A}$	the hash value that should be *passed to* participant A
SIG_A	the signature generated by participant A
$s_{i_{\to A}}$	the ith shared signature that should be *passed to* participant A
T_e	the timestamp when the purchase request expires
T_{i_A}	the ith timestamp at participant A
TR_A	the transaction record kept by participant A
(y_{K_A}, x_{K_A})	(*public key, secret key*) of participant A for encryption and decryption
(y_{S_A}, x_{S_A})	signature ($public\ key,\ secret\ key$) of participant A
z	a random number chosen from $[1, \ldots, q]$
$X\|\|Y$	concatenation of two messages X and Y
$A \to B : m$	participant A sends a message m to participant B

Table 1. Notations and Terms

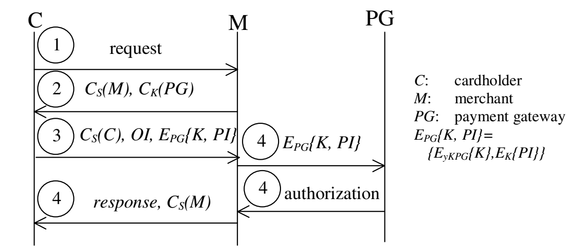

Figure 1. SET Purchase Request Transaction

3. Proposed Protocol

In the proposed protocol, Signature-Share scheme is adopted for passing securely the order information to the merchant while Signcryption-Share scheme is adopted for passing the payment information (PI) to the payment gateway (PG) and a temporary session public key pair is used to encrypt PI. The cardholder's signature private key is divided into two parts. The first part is kept by the cardholder. The second part is encrypted by the public key of the TTP and will be passed to the TTP for generating shared signatures. The dispatched agent does not carry any shared signature private key. Instead it only carries two half shared signatures signed on the order information (OI_j) and PI respectively by the cardholder that should be sent to the merchant M_j and payment gateway PG_j, which is chosen by M_j according to the card brand. With the same brand, different merchants may have different payment gateways from different acquirers. The other 2 half shared signatures are generated with the assistance of the TTP. On obtaining the two shared signatures (i.e. $s_{1_{\to M_j}}$ and $s_{2_{\to M_j}}$), the merchant M_j can verify OI_j (for the jth product) and check the data integrity. Meanwhile PG_j can not only decrypt PI but also check the data integrity after obtaining its two shared signatures (i.e. $s_{1_{\to PG_j}}$ and $s_{2_{\to PG_j}}$).

3.1. Secret-Sharing of Cardholder's Signature Private Key x_{S_C}

In the proposed protocol, the cardholder and TTP share the cardholder's signature private key x_{S_C} based on shamir-threshold scheme [8].

$$x_{S_C} = x_{S_{C_1}} + x_{S_{TTP}}$$

Namely, according to the two share schemes, $A_1 = C$ and $A_2 = TTP$. $x_{S_{C_1}}$ is kept by C as a secret key always while $x_{S_{TTP}}$ can be carried by the agent after being encrypted using the TTP's public key and will be passed to the TTP for generating the second shared signatures that will be passed to M_j and PG_j respectively.

3.2. Description of Proposed Protocol

To describe our proposed protocol, for the sake of simplicity, it is assumed that the agent will buy N products from N merchants. $Transaction_j$ means the transaction with merchant M_j selling $Product_j$. Two products may be bought from the same merchant in different transactions.

Step 1: Cardholder C generates a pair of temporary session keys -$(K_{y_{PG_j}}, K_{x_{PG_j}})$, where $K_{y_{PG_j}} = g^{K_{x_{PG_j}}} \ mod \ p$, for the payment gateway. It is different from PG_j's encryption public key pair -$(y_{K_{PG_j}}, x_{K_{PG_j}})$.

1) Then C uses Signcryption algorithm to encrypt the payment information (PI):

$$(k_1, k_2) = H({K_{y_{PG_j}}}^z \ mod \ p)$$
$$c_{\to PG_j} = E_{k_1}\{PI\}$$

generate the hash value:

$$r_{\to PG_j} = KH_{k_2}\{PI\}$$

generate the first half shared signature to PG_j:

$$s_{1_{\to PG_j}} = z/(r_{\to PG_j} + x_{S_{C_1}}) \ mod \ q$$

and generate the ciphertext

$$E_{y_{K_{TTP}}}\{x_{S_{TTP}}||z||(K_{x_{PG_j}} + R_j + I_C + T_C + T_e)\}.$$

where

- R_j is a random number chosen from $[1, \ldots, q]$;
- I_C is the transaction identifier assigned by cardholder C and T_C is the timestamp at C when to complete the encryption and shared signature generation;
- T_e ($T_e > T_C$) is the timestamp when the purchase request expires. It is unique to each purchase order.

Note: $s_{1_{\to PG_j}}$ is the half shared signature generated by C that should be passed to payment gateway PG_j and the consumer agent carries it instead of the shared secret key- $x_{S_{C_1}}$. $x_{S_{C_1}}$ is kept by C.

2) Meanwhile, C generates the first half shared signature $s_{1_{\to M_j}}$ on the dual hash value that will be passed to the merchant M_j:

$$r_{\to M_j} = H(g^z \ mod \ p, H(PI)||H(OI_j)|| H(C_S(C)||I_C||T_C||T_e))$$
$$s_{1_{\to M_j}} = z/(r_{\to M_j} + x_{S_{C_1}}) \ mod \ q$$

where

- OI_j is the description and constraint for the order of $Product_j$, namely, $OI_j = OrderDescription_j,\ PriceLimit_j,$ $x_{S_C}(H(OrderDescription_j,$ $PriceLimit_j,\ T_C))$

Proceedings of the 2005 IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'05)
0-7695-2416-8/05 $20.00 © 2005 IEEE

3) Then C dispatches the consumer agent CA encapsulating the following arguments:

$C_S(C),\ C_K(C), \{E_{y_{K_{TTP}}}\{x_{S_{TTP}}||z||(K_{x_{PG_j}}+ R_j+I_C+T_C+T_e)\}\},\ OI,\ H(PI),\ R,\ I_C,\ T_C,\ T_e,\ r_{\to M},\ s_{1_{\to M}},\ c_{\to PG},\ r_{\to PG},\ s_{1_{\to PG}},\ SIG_C$

where

- $OI = \{OI_j | j = 1,\ \ldots,\ N\}$
- $R = \{R_j | j = 1,\ \ldots,\ N\}$
- $r_{\to M}$=$\{r_{\to M_j} | j = 1,\ \ldots,\ N\}$
- $s_{1_{\to M}}$=$\{s_{1_{\to M_j}} | j = 1,\ \ldots,\ N\}$
- $c_{\to PG} = \{c_{\to PG_j} | j = 1,\ \ldots,\ N\}$
- $r_{\to PG} = \{r_{\to PG_j} | j = 1,\ \ldots,\ N\}$
- $s_{1_{\to PG}} = \{s_{1_{\to PG_j}} | j = 1,\ \ldots,\ N\}$
- $SIG_C = x_{S_C}(H(C_S(C),\ C_K(C),\ \{E_{y_{K_{TTP}}}\{ x_{S_{TTP}}||z||(K_{x_{PG_j}} + R_j + I_C + T_C + T_e)\}\},\ OI,\ H(PI),\ R,\ I_C,\ T_C,\ T_e,\ r_{\to M},\ s_{1_{\to M}},\ c_{\to PG},\ r_{\to PG},\ s_{1_{\to PG}}))$

The dispatched agent will visit a set of merchants asking offers and negotiating with them [14].

Step 2: After completing the negotiation with merchants, the agent selects merchant M_j, which is the best merchant with the best offer for $product_j$, to make the deal and send M_j the purchase request. The request includes the brand of the credit card that will be used for payment.

$CA \to M_j :\ C_S(C),\ purchase\ request,\ T_e$

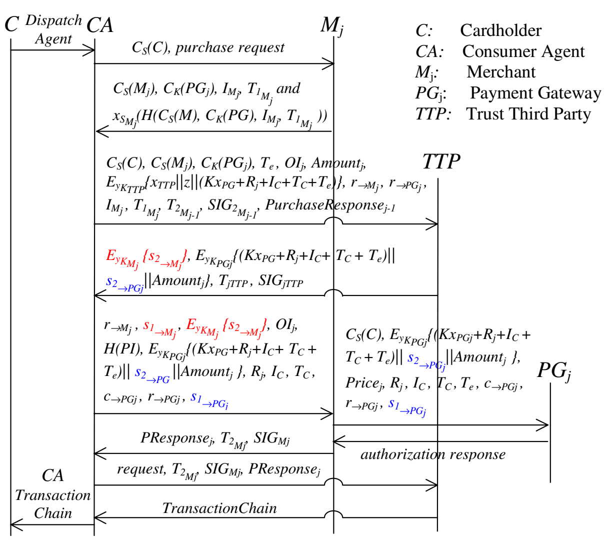

Figure 2. Purchase Request Transaction

Step 3: After receiving the request, M_j verifies $C_S(C)$ and reply CA.

$$M_j \to CA :\ C_S(M_j),\ C_K(PG_j),\ I_{M_j},\ T_{1_{M_j}}\ and\ x_{S_{M_j}}(H(C_S(M_j),\ C_K(PG_j),\ I_{M_j},\ T_{1_{M_j}}))$$

where

- I_{M_j} is a unique transaction number issued by M_j and $T_{1_{M_j}}$ is the current timestamp at M_j;
- $x_{S_{M_j}}(H(C_S(M_j),\ C_K(PG_j),\ I_{M_j},\ T_{1_{M_j}}))$ is the signature generated by M_j.

Step 4: From M_j's reply, CA obtains the public key certificate of the payment gateway PG_j. Then CA sends TTP a message so that $s_{2_{\to PG_j}}$ and $s_{2_{\to M_j}}$ can be generated.

$$CA \to TTP :\ C_S(C),\ C_S(M_j),\ C_K(PG_j),\ T_e,\ OI_j,\ Amount_j,\ E_{y_{K_{TTP}}}\{x_{S_{TTP}}||z||(K_{x_{PG_j}}+R_j+I_C+T_C+T_e)\},\ r_{\to M_j},\ r_{\to PG_j},\ I_{M_j},\ T_{1_{M_j}},\ T_{2_{M_{j-1}}},\ SIG_{M_{j-1}},\ PRresponse_{j-1}$$

where

- $Amount_j$=$Price_j$. $Price_j$ is the price of $Product_j$, which is determined by CA and M_j. Here we distinguish $Amount_j$ and $Price_j$ as both of them will be passed to the PG_j where a consistency check will be performed;
- $T_{2_{M_{j-1}}}$ is the second timestamp at M_{j-1} (given in Step 9);
- $SIG_{M_{j-1}}$ is the signature of M_{j-1} (given in Step 9);
- $PResponse_{j-1}$ is the purchase response from M_{j-1} (given in Step 9).

Step 5: On receiving the message, TTP verifies the validation of $C_S(C)$, $C_S(M_j)$, $C_K(M_j)$ and $C_K(PG_j)$, checks whether the current time $T < T_e$ and $Amount_j \leq PriceLimit_j$. If all are correct, TTP decrypts the ciphertext from CA obtaining z and $x_{S_{TTP}}$, generates 2 half shared signatures on hash values $r_{\to PG_j}$ and $r_{\to M_j}$ respectively,

$$s_{2_{\to PG_j}} = z/(r_{\to PG_j} + x_{S_{TTP}})\ mod\ q$$
$$s_{2_{\to M_j}} = z/(r_{\to M_j} + x_{S_{TTP}})\ mod\ q$$

and generates

$$E_{y_{K_{PG_j}}}\{(K_{x_{PG_j}} + R_j + I_C + T_C + T_e\}|| s_{2_{\to PG_j}}||Amount_j\}$$ and $E_{y_{K_{M_j}}}\{s_{2_{\to M_j}}\}$

Note: TTP knows $(K_{x_{PG_j}} + R_j + I_C + T_C + T_e)$ but doesn't know $K_{x_{PG_j}}$.

Hereafter TTP keeps

$$TR_{j_{TTP}} = \{C_S(C),\ C_S(M_j),\ I_C,\ T_C,\ T_e,\ I_{M_j},\ T_{1_{M_j}},\ T_{TTP},\ x_{S_{M_j}}(H(C_S(M_j),\ C_K(PG_j),\ I_{M_j},\ T_{1_{M_j}}))\}$$

as a transaction record and sends a message to CA.

Proceedings of the 2005 IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'05)
0-7695-2416-8/05 $20.00 © 2005 IEEE

$TTP \rightarrow CA: E_{y_{K_{M_j}}}\{s_{2\rightarrow M_j}\},\ E_{y_{K_{PG_j}}}\{(K_{x_{PG_j}} + R_j + I_C + T_C + T_e)||s_{2\rightarrow PG_j}||Amount_j\},\ T_{j_{TTP}},\ SIG_{j_{TTP}}$

where

- $T_{j_{TTP}}$ is the timestamp at TTP when to generate the shared signatures (i.e. $s_{2\rightarrow PG_j}$ and $s_{2\rightarrow M_j}$);
- $SIG_{j_{TTP}} = x_{S_{TTP}}(H(OI_j,\ C_S(C),\ C_S(M_j),\ C_K(PG_j),\ E_{y_{K_{M_j}}}\{s_{2\rightarrow M_j}\},\ E_{y_{K_{PG_j}}}\{(K_{x_{PG_j}} + R_j + I_C + T_C + T_e)||s_{2\rightarrow PG_j}||Amount_j\},\ r_{\rightarrow M_j},\ r_{\rightarrow PG_j},\ I_{M_j},\ T_{j_{TTP}}))$ is the signature generated by TTP for $Transaction_j$ that can be kept by the cardholder as a non-repudiation receipt.

Step 6: Once receiving the message from TTP, CA sends a message to the merchant.

$CA \rightarrow M_j: r_{\rightarrow M_j},\ s_{1\rightarrow M_j},\ E_{y_{K_{M_j}}}\{s_{2\rightarrow M_j}\},\ OI_j,\ H(PI),\ E_{y_{K_{PG_j}}}\{(K_{x_{PG_j}} + R_j + I_C + T_C + T_e)||s_{2\rightarrow PG_j}||Amount_j\},\ R_j,\ I_C,\ T_C,\ c_{\rightarrow PG_j},\ r_{\rightarrow PG_j},\ s_{1\rightarrow PG_j}$

Step 7: After having received the message, M_j computes v_j by applying the Signature-Share scheme

$$v_j = H((y_{K_C} \cdot g^{2r})^{(\sum_{i=1}^{2} s_{1\rightarrow M_j}{}^{-1})^{-1}}\ mod\ p)$$

and verify signature

$$H(v_j, H(PI)||H(OI_j)||H(C_S(C)||I_C||T_C||T_e)) \stackrel{?}{=} r_{\rightarrow M_j}$$

If it holds and the current time $T < T_e$, M_j keeps $TR_{M_j} = \{C_S(C),\ r_{\rightarrow M_j},\ s_{1\rightarrow M_j},\ s_{2\rightarrow M_j},\ OI_j,\ H(PI),\ I_C,\ T_C,\ T_e\}$ as a transaction record.

Then M_j sends a message to PG_j.

$M_j \rightarrow PG_j: C_S(C),\ E_{y_{K_{PG_j}}}\{(K_{x_{PG_j}} + R_j + I_C + T_C + T_e)||s_{2\rightarrow PG_j}||Amount_j\},\ Price_j,\ R_j,\ I_C,\ T_C,\ T_e,\ c_{\rightarrow PG_j}, r_{\rightarrow PG_j}, s_{1\rightarrow PG_j}$

Step 8: From the message, PG_j obtains $s_{1\rightarrow PG_j}$. After decrypting $E_{y_{K_{PG_j}}}\{(K_{x_{PG_j}} + R_j + I_C + T_C + T_e)||s_{2\rightarrow PG_j}||Amount_j\}$, it obtains $K_{x_{PG_j}}$, $s_{2\rightarrow PG_j}$ and $Amount_j$. Hereafter PG_j can apply the Signcryption-Share scheme to decrypt $c_{\rightarrow PG_j}$ and thus obtain PI:

$$(k_1, k_2) = H((y_{K_C} \cdot g^{2r})^{(\sum_{i=1}^{2} s_{i\rightarrow PG_j}{}^{-1})^{-1} \cdot K_{x_{PG_j}}}\ mod\ p)$$

$$PI = D_{k_1}\{c_{\rightarrow PG_j}\}$$

and check data integrity:

$$KH_{k_2}\{PI\} \stackrel{?}{=} r_{\rightarrow PG_j}$$

If it holds, the current time $T < T_e$, and $Amount_j = Price_j$, PG_j contacts the card issuer for authorizing the payment. Hereafter, PG_j sends M_j an authorization response.

Step 9: After processing the order, the merchant generates and signs a purchase response $PResponse_j$, and sends it to the agent.

$M_j \rightarrow CA: PResponse_j,\ T_{2_{M_j}}, SIG_{M_j}$

where

- $T_{2_{M_j}}$ is the timestamp ($T_{2_{M_j}} > T_{1_{M_j}}$) at M_j when SIG_{M_j} is issued;
- $SIG_{M_j} = x_{S_{M_j}}(H(PResponse_j,\ r_{\rightarrow M_j},\ s_{1\rightarrow M_j},\ s_{2\rightarrow M_j},\ R_j,\ OI_j,\ H(PI),\ I_C,\ T_C,\ T_e,\ I_{M_j},\ T_{2_{M_j}}))$ is the signature generated by M_j at time $T_{2_{M_j}}$. It will be finally passed to the cardholder as a non-repudiation receipt by the agent.

If the payment is authorized, M_j fulfills the order by delivering the product bought by the cardholder.

Step 10: The agent checks the digital signature of the response and the merchant's signature. If there are other transactions, CA will communicate with merchant M_{j+1} and repeat Steps 2-9. If this is the last transaction, CA sends TTP a request asking for generating the transaction chain:

$CA \rightarrow TTP: request,\ T_{2_{M_j}},\ SIG_{M_j},\ PResponse_j,\ C_K(C)$

Step 11: Once receiving CA's request, TTP generates a session symmetric key $K_{\rightarrow C}$, encrypt $K_{\rightarrow C}$ in a digital envelope to C and use $K_{\rightarrow C}$ to encrypt the $Element_N, \ldots, Element_1$ in order forming a transaction chain and transmit it to CA.

$TTP \rightarrow CA: TransactionChain$

where

- $TransactionChain = Element_1||Element_2||\ldots||Element_N||T_{N+1_{TTP}}||E_{y_{K_C}}\{K_{\rightarrow C}\}$
- $Element_i$ includes the transaction record for $Transaction_j$
- $E_j = E_{K_{\rightarrow C}}\{Amout_j,\ PResponse_j,\ T_{1_{M_j}},\ T_{2_{M_j}},\ I_{M_j},\ T_{j_{TTP}},\ E_{y_{K_{PG_j}}}\{K_{x_{PG_j}} + R_j + I_C + T_C + T_e\},\ SIG_{M_j}\}$
- $Element_i = E_j,\ x_{S_{TTP}}(H(E_{y_{K_C}}\{K_{\rightarrow C}\},\ C_S(M_j),\ C_K(PG_j),\ E_j,\ Element_{j+1},\ T_{N+1_{TTP}}))$
- $T_{N+1_{TTP}}$ is the $(N+1)th$ timestamp at TTP when $TransactionChain$ is generated.
- $Element_{N+1} = NULL$

Step 12: CA then returns back to its owner carrying $C_S(TTP)$ and $TransactionChain$. The owner takes appropriate actions based on the obtained contents.

Here we describe an iterative process of the protocol. For simplicity, the agent can transmit all relevant information to the TTP after having found N merchants. The TTP then generates all shared signatures and pass them to CA within one interaction only.

4. Security Analysis

In this section, we analyze the security properties of the proposed protocol focusing on the following possible issues in two categories.
Category 1 (Security Properties of Each Transaction):

- whether it is possible for any participant to re-generate the secret signature key of the cardholder (ATK1);
- whether it is possible for any participant except PG_j to obtain the payment information (ATK2);
- whether it is possible for any participant to re-perform the payment (double payment, ATK3);
- whether it is possible for the agent pay more than required (overspend, ATK4)
- whether it is possible for the merchant to pass a wrong price to the PG (over payment, ATK5).

In the proposed protocol, the dispatched agent CA does not have any task for encryption, decryption or signing. So it is not necessary for it to carry any keys. In the proposed protocol, the agent in the transaction is more of a messenger. Most of the encryption and signing work are done by the TTP. What the agent should do is to communicate with different participants sending relevant messages to them.

1. CA carries two shared half signatures -$s_{1_{\to M}}$ and $s_{1_{\to PG_j}}$. But they are generated by cardholder C and the shared secret key $x_{S_{C_1}}$ is kept by C. No party could obtain both two shared signatures (i.e. $s_{1_{\to M}}$ and $s_{2_{\to M}}$, or $s_{1_{\to PG_j}}$ and $s_{2_{\to PG_j}}$) together with some argument (i.e. r and z); so it is not possible for any party to obtain two shared secret keys so as to generate the secret signature key of the cardholder (i.e. x_{S_C}) (ATK1).

   For instance, for the merchant, it can obtain the $r_{\to M}$, $s_{\to M} - 1$, $s_{2_{\to M}}$, $c_{\to PG_j}$, $r_{\to PG_j}$, $s_{\to PG_j} - 1$ and $H(PI)$, but cannot obtain PI and $s_{2_{\to PG_j}}$. Argument z is also protected against each merchant. So it is not possible for M_j to obtain x_{S_C} (see Figure 3).

   Likewise, TTP knows $(K_{xPG_j}+R_j+I_C+T_C+T_e)$ but doesn't know K_{xPG_j}. Meanwhile $E_{k_1}\{PI\}$ is not passed to TTP. As $s_{1_{\to M}}$ and $s_{1_{\to PG_j}}$ are not passed to TTP, TTP cannot generate $x_{S_{C_1}}$ so as to re-generate x_{S_C} (see Figure 4).

   In the proposed protocol, the cardholder's secret signature key can be re-generated only if M and TTP collude. But it is impossible regarding the nature of TTP.

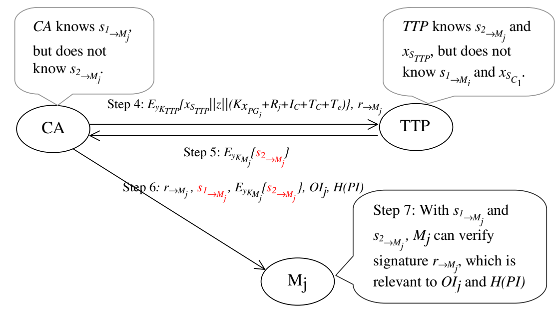

Figure 3. Shares Passed to M_j

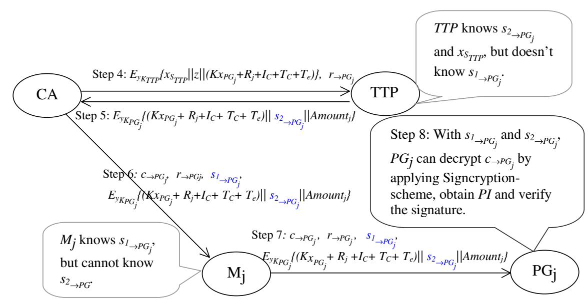

Figure 4. Shares Passed to PG_j

2. After obtaining 2 shared signatures -$s_{1_{\to PG_j}}$ and $s_{2_{\to PG_j}}$- PG_j can not only decrypt the payment information PI but also check the data integrity. Its session secret key K_{xPG_j} is encrypted as $E_{yK_{PG_j}}\{(K_{xPG_j} + R_j + I_C + T_C)||s_{2_{\to PG_j}}\}$. M_j knows the ciphertext but doesn't know $y_{K_{PG_j}}$ and $s_{2_{\to PG_j}}$ (ATK2).
3. In our proposed protocol, as each payment is identified by T_C, I_C together with the signatures of C, the replayed payment can be detected by PG (ATK3).
4. In our proposed protocol, $Amount_j$, the amount of the transaction that will be charged to the cardholder's account is first passed to the TTP, which checks it with the limit of current transaction (i.e. $PriceLimit_j$). Moreover, the amount is included in the ciphertext by the TTP that will be passed to the PG_j where the comparison will be conducted with the price (i.e. $Price_j$) from M_j. This can prevent the overspending and over payment attacks (ATK4&ATK5).

Category 2: (Security Properties of Multiple Payments and Transactions):

- whether it is possible to disclose the transaction information to other merchants (ATK6);
- whether it is possible for any participant to insert data to transaction records (ATK7);
- whether it is possible for any participant to modify or delete data in transaction records (ATK8).

1. The transaction information (e.g. $Amount_j$) is passed to TTP in each transaction. Only TTP and M_j know it. It is not exposed to other merchants (ATK6).
2. Each transaction record (i.e. $Element_j$) is encrypted by the session symmetric key $K_{\rightarrow C}$ while $Element_{j+1}$ appears in the signature in $Element_j$ forming a transaction chain including transaction records and timestamps in each transaction. This structure sets up the dependency between adjacent elements. Deleting any of them can be detected by C (ATK8). Meanwhile, as each element is signed by TTP, it is not possible for other participant to forge or insert a new element (ATK7).

5. Conclusions

In this paper, we proposed an agent-assisted secure payment protocol supporting multiple payments, which adopts Signature-Share scheme and Signcryption-Share scheme and employs a Trusted Third Party (TTP). In the proposed protocol, the principle that each participant knows what is strictly necessary for his/her role is followed as in SET while the non-repudiation property is improved. The dispatched agent can dynamically and flexibility choose the merchant and sign on behalf of the cardholder in cooperation with the TTP without the possibility of disclosing any secret credit card's information to the merchants and TTP. Offers' information is protected against irrelevant merchants.

To reduce the risk of employing mobile agents, the reputation and trust status of merchants can be evaluated in advance. We ever proposed relevant models in [14] and [6]. For future work, we will integrate the proposed protocol into our PumaMart system: an agent-mediated B2C Internet marketplace system [13, 14] implemented on top of Java and IBM Aglets toolkits [5].

References

[1] *Visa International and MasterCard International*. Secure Electronic Transaction (SET) specification, Version 1.0, May 1997.

[2] R. M. A. Corradi and C. Stefanell. Mobile agent integrity in e-commerce application. In *Proceedings of 19th IEEE International Conference on Distributed Computing Systems*, pages 59–64, 1999.

[3] D. Chess. Security issues in mobile code systems. In *Proceedings of Mobile Agents and Security*, pages 1–14, 1998. Springer Verlag, LNCS 1419.

[4] R. Guttman and P. Maes. Agent-mediated integrative negotiation for retail electronic commerce. In *Proceedings of the Workshop on Agent Mediated Electronic Trading (AMET'98)*, pages 1–13, 1998.

[5] D. B. Lange and M. Oshima. *Programming and Deploying Java Mobile Agents with Aglets*. Addison-Wesley Press, Massachusetts.

[6] C. Lin, V. Varadharajan, and Y. Wang. On the design of a new trust model for mobile agent security. In *Proceedings of 1st International Conference on Trust and Privacy in Digital Business (TrustBus04)*, volume LNCS 3184, pages 60–69, Zaragoza, Spain, August-September 2004.

[7] P. Maes, R. Guttman, and A. Moukas. Agents that buy and sell. *CACM*, 42(3):81–91, 1999.

[8] A. Menezes, P. Oorschot, and S. Vanstone. *Handbook of applied cryptography*. CRC Press, 1996.

[9] X. Pang, K.-L. Tan, and Y. Wang. A secure agent-mediated payment protocol. In *Fourth International Conference on Information and Communications Security (ICICS2002)*, volume LNCS 2512, pages 422–433, Singapore, December 2002. Springer-Verlag.

[10] R. L. Rivest and A. Shamir. Payword and micromint: Two simple micropayment schemes. In *CryptoBytes*, pages 7–11, 1996.

[11] A. Romao and M. M. da Silva. An agent-based secure internet payment system for mobile computing. In *Proceedings of TrEC'98*, Hamburg, Germany, 1998. Springer.

[12] Y. Wang and T. Li. LITESET/A++: A new agent-assisted secure payment protocol. In *Proceedings of 6th IEEE International Conference on E-Commerce Technology (IEEE CEC'04)*, San Diego, California, USA, July 2004. IEEE Computer Society.

[13] Y. Wang, K.-L. Tan, and J. Ren. A study of building internet marketplaces on the basis of mobile agents for parallel processing. *World Wide Web Journal, Kluwer Academics Publisher*, 5(1):41–66, 2002.

[14] Y. Wang, K.-L. Tan, and J. Ren. Pumamart: A parallel and autonomous agents based internet marketplace. *Electronic Commerce Research and Applications*, 3(3):294, 2004.

[15] X. Yi, C. K. Siew, X. F. Wang, and E. Okamoto. A secure agent-based framework for the internet trading in mobile computing environments. *Distributed and Parallel Databases*, 8:85–117, 2000.

[16] Y. Zheng. Digital signcryption or how to achieve cost (signature and encryption) $\ll$ cost (signature)+cost (encryption). In *Proceedings of Advances in Cryptology-CRYPO'97*, volume 1294, pages 165–179. Springer-Verlag, 1997.