Research on Mobile Multi-bank Divisible e-Cash Protocol

Ting Huang

Abstract—Mobile multi-bank divisible e-cash protocol proposes on mobile terminals for electronic payments which can circulate in multiple banks and cannot limit from the bank that issues the e-cash. This paper proposes on the agreement of opening account, the withdrawal agreement, the pay agreement, the deposit agreement, update protocol of the e-cash based on elliptic curve cryptography (ECC). The divisible e-cash does not pass the bank by the consumption of each e-cash, thus the bank has not become the bottleneck of electronic cash transactions. The protocol is more suitable for mobile micro-payment terminals with limited calculation capacity, storage, network bandwidth and power supply, which satisfies the needs of day-to-day transactions.

Index Terms—Mobile, multi-bank, micro-payment, divisible e-cash.

I. INTRODUCTION

Along with improvement of the global information technology, electronic commerce has become an important business model in twenty-first Century, and showed a very important role in the national economy. However, e-cash systems use the online payment now. The consumption of e-cash must pass the bank [1], thus the bank has become the bottleneck of electronic cash transactions. The e-cash of literature [2] only circulates in a bank, which cannot meet the needs of reality. The e-cash protocol [3] bases on RSA problem that need exponentiation compute. The e-cash [4] cannot circulate in merchant’s commerce. This paper proposes on mobile, safe, efficient and multi-bank e-cash protocol. The research effectively improves the overall efficiency of mobile e-cash system and the ability of instant payment, can enhance the security and practicability of electronic cash system, and provides the theoretical basis and technical support for a transaction system of electronic commerce.

II. K-ARY TREE NODES OF ELECTRONIC CASH CAN TAKE ALL

In the agreement I propose that each node of the K-ary tree can be cost by users. When a node of K-ary tree can meet the user’s spending needs, the user can directly take a node or some nodes of K-ary tree by polymerization. When the nodes of K-ary tree cannot satisfy the user’s spending needs, the users can subdivide a leaf node of the K-ary tree that cannot be used again. That is a n layer of K-ary tree. The fork root node of K-ary tree represents the amount of , while the total amount of the entire classical K-ary tree is and the total amount of K-ary tree in this agreement is . The classical principle requires that the user withdraws money times, the bank signs times for the user. But now the user only need to withdraw 1 time, and the bank only need to sign 1 times for the user. When the e-cash deposits to the bank, the classical principle needs that the bank do operations on nodes of K-ary tree while the bank only need do operations on nodes of a K-ary tree in this paper. So from the signature number of e-cash in the withdrawal protocol and detecting double spending in the deposit agreement the new principles is just of classic K-ary tree, which improves the efficiency of divisible e-cash protocol. For example, a three layer of the 10-ary tree is shown in Fig. 1.

III. DESIGN OF AGREEMENT

The point of abscissa and vertical axis are x and y on the elliptic curve respectively; are the user, credit center, factory, merchant’s private key; are the user, credit center, factory, merchant’s public key. The user’s name is . Each bank is classified by a number, such as: CCB is the bank, …; another bank is the bank, of which the corresponding private key and the signature private key are defined as , and the corresponding public key is .

Fig. 1. Three layer of the 10-ary tree.

TC is defined as the use cycle of the e-cash (TC is the time cycle for ). generates when the user gets the e-cash from the bank. The time cycle for is an effective use cycle of the e-cash (TC’ is the given value). When reaches the e-cash is unavailable. The user must go to the bank for updating the e-cash only before . (The time cycle for is the updating cycle of the e-cash that is the given value).

A. Withdrawal Agreement

Step 1: The user fetches Q yuan from the bank and gets the segmentation parameter k of the e-cash. Then he gets . Then he gets the k-ary tree by Q yuan, saves , after encrypting .
computes
\[ \alpha_{\text{na}} = H[(\text{QSK}_{\text{Bel}} + p\text{SK}_{\text{Bel}})] \| \text{SK}_{\text{Bal}} \| T_1 \| \| B] \]
and
\[ \beta_{(n)} = H[(\text{CG})_a \| \alpha_{\text{na}} \| T_1 \| Q]. \]

The bank \( \alpha_{\text{na}} \) saves them to withdrawal database.

**B. Payment Agreement**

The user pays the e-cash to the merchant, through the credit center. The payments in two ways are described in Table I and Table II. The courses that merchant pays the e-cash to Factory or Merchant, \( (i = 1, 2, \ldots, n) \) are described in Table III and Table IV.

### Table I: Payment Agreement (Q=Q Yuan)

<table>
<thead>
<tr>
<th>User</th>
<th>Credit Center</th>
<th>Merchant</th>
</tr>
</thead>
<tbody>
<tr>
<td>User obtains the merchant's address</td>
<td>It sends G, NA, Timestamp and notices merchant send goods</td>
<td>It sends G, NA, Timestamp and notices merchant send goods</td>
</tr>
<tr>
<td>Q, NA, Timestamp, merchant's address</td>
<td>When he gets the goods, the user computes</td>
<td>It receives G = H[Q</td>
</tr>
<tr>
<td>G = H[Q</td>
<td></td>
<td>SK_{\text{Bal}}]]</td>
</tr>
<tr>
<td>It sends G, NA, Timestamp and notices merchant send goods</td>
<td>SAVES Q, NA, Timestamp, merchant's address</td>
<td>The merchant exams</td>
</tr>
</tbody>
</table>

### Table II: Payment Agreement (Q\(<Q\) Yuan)

<table>
<thead>
<tr>
<th>User</th>
<th>Credit Center</th>
<th>Merchant</th>
</tr>
</thead>
<tbody>
<tr>
<td>User obtains the merchant's address</td>
<td>It sends G, NA, Timestamp and notices merchant send goods</td>
<td>It sends G, NA, Timestamp and notices merchant send goods</td>
</tr>
<tr>
<td>Q, NA, Timestamp, merchant's address</td>
<td>When he gets the goods, the user computes</td>
<td>It receives G = H[Q</td>
</tr>
<tr>
<td>G = H[Q</td>
<td></td>
<td>SK_{\text{Bal}}]]</td>
</tr>
<tr>
<td>It sends G, NA, Timestamp and notices merchant send goods</td>
<td>SAVES Q, NA, Timestamp, merchant's address</td>
<td>The merchant exams</td>
</tr>
</tbody>
</table>

### Table III: Merchant's Payment Agreement (Q\(<Q\) Yuan)

<table>
<thead>
<tr>
<th>Merchant</th>
<th>Credit Center</th>
<th>Merchant</th>
</tr>
</thead>
<tbody>
<tr>
<td>Merchant receives</td>
<td>It computes G = H[Q</td>
<td></td>
</tr>
<tr>
<td>Q, NA, merchant's address</td>
<td>Sends G, NA, Timestamp and notices merchant send goods,</td>
<td>The merchant exams</td>
</tr>
<tr>
<td>Q, NA, merchant's address</td>
<td>It computes G = H[Q</td>
<td></td>
</tr>
<tr>
<td>After he gets the goods, the merchant computes</td>
<td>It computes G = H[Q</td>
<td></td>
</tr>
<tr>
<td>F = H[SK_{\text{Bal}}]]</td>
<td>SAVES Q, NA, Timestamp, merchant's address</td>
<td>The merchant exams</td>
</tr>
<tr>
<td>Q, It sends Timestamp, F.</td>
<td>SAVES Q, NA, merchant's address</td>
<td>The merchant exams</td>
</tr>
<tr>
<td>NA, merchant's address</td>
<td>SAVES Q, NA, Timestamp, merchant's address</td>
<td>The merchant exams</td>
</tr>
</tbody>
</table>

438
C. Deposit Agreement

After the merchant sends the relevant transaction information that stored in the payment protocol to the bank, the bank verifies the correctness of the e-cash to detect double-spending. When the merchant deposits to the bank, the e-cash will deposit to the bank, the e-cash through the withdrawal database. Otherwise, the bank will get (q', q) from the person's signature, and does the penalty.

In this section the payment node sent by the business as an example: The e-cash is checked whether it has the same path p, you can conclude the reuse of the e-cash. Then the example: The e-cash is checked whether it has the same path p, no transfers, and does the penalty.

Based on blind signature protocol with higher level than ordinary bank which verifies the authenticity of the transfer e-cash on a regular time, detects the false transfers, and does the penalty.

D. Update of the e-Cash

Step 1: The user generates A ∈ R Zn, ρ = H(A + SKu), regenerates the K tree by the residual e-cash Q, and saves ρ, K to the database. After they are encrypted by the bank, the user’s shared key, ρ, K are sent to the bank.

Step 2: The bank regenerates the root node of the divisible e-cash according to the user’s new update time T:

\[
\alpha_n = H[(u_{QSK_{bas}} + pSK_{bas}) || SK_{bas} || T || B],
\]

computes

\[
\beta_n = H[(CG) || \alpha_n || T || Q],
\]

\[
\gamma_n = C + \beta_nSK_{bas},
\]

C ∈ R Zn, and saves \(\alpha_n, \beta_n, \gamma_n, T\) to the database. ID_{db}.

\[
\alpha_n, \beta_n, \gamma_n, Q, T\] are sent to the user after they are encrypted.

The e-cash can be transacted by previous protocol after it has been updated. The same course happens in the e-cash:

\[
\alpha_{n1} || \ldots || \alpha_{nk} \cdot H_{0(n) || \ldots || h_n} \cdot K_{0(n) || \ldots || k_n}.
\]

IV. ANALYSIS OF SECURITY AND EFFICIENCY

Based on blind signature protocol with higher level than ordinary bank which verifies the authenticity of the transfer e-cash on a regular time, detects the false transfers, and does the penalty.

In this section the payment node sent by the business as an example: The e-cash is checked whether it has the same path p, no transfers, and does the penalty.

D. Update of the e-Cash

Step 1: The user generates A ∈ R Zn, ρ = H(A + SKu), regenerates the K tree by the residual e-cash Q, and saves ρ, K to the database. After they are encrypted by the bank, the user’s shared key, ρ, K are sent to the bank.

Step 2: The bank regenerates the root node of the divisible e-cash according to the user’s new update time T:

\[
\alpha_n = H[(u_{QSK_{bas}} + pSK_{bas}) || SK_{bas} || T || B],
\]

computes

\[
\beta_n = H[(CG) || \alpha_n || T || Q],
\]

\[
\gamma_n = C + \beta_nSK_{bas},
\]

C ∈ R Zn, and saves \(\alpha_n, \beta_n, \gamma_n, T\) to the database. ID_{db}.

\[
\alpha_n, \beta_n, \gamma_n, Q, T\] are sent to the user after they are encrypted.

The e-cash can be transacted by previous protocol after it has been updated. The same course happens in the e-cash:

\[
\alpha_{n1} || \ldots || \alpha_{nk} \cdot H_{0(n) || \ldots || h_n} \cdot K_{0(n) || \ldots || k_n}.
\]
The time of protocol implementation and storage consumption in mobile multi-bank divisible e-cash protocol are the key to efficiency. 160 bit length of the key in ECC has the same powerful function as 1024 bits length of the key in RSA [5]. The e-cash protocol [3] bases on RSA problem. The agreement of this system bases on ECC. Compared to RSA, ECC needn’t exponentiation compute. The computation of ECC is negligible compared to RSA. Therefore the efficiency of my protocol runs faster. Paying a e-cash [3] needs to save $C_i(2^i.S.T,\Phi,R,\overrightarrow{V}_i,\overrightarrow{V}_2,\overrightarrow{V}_3)(i=3)$. The storage space is at least $1024^9=9216$bit=1152byte. While paying a e-cash in my paper simply saves $Q',\alpha_{(n)11,12,13} \cdot \beta_{(n)11,12,13} \cdot \gamma_{(n)11,12,13} \cdot p=112,12,13$. K (Assuming the K-tree assigns to the third branch) whose storage space is approximately $32+128+128+192+128+32=736$bit=92byte, which reduces at least 92% of the storage amount and the network bandwidth. The two-tree is used in [3]. Because the branches in splitting the e-cash are much more than the K-tree, the amount of computation in generating the branches of the e-cash is increased too.

The e-cash in my paper contains the use cycle. When the end of the use cycle comes, the bank will delete the e-cash. Thus much storage space is saved.

The multi-bank e-cash [6] requires the central bank to issue the e-cash. A financial institution is added, which lets the e-cash is increased too. The amount of computation in generating the branches of the e-cash are much more than the K-tree, network bandwidth. The two-tree is used in [3]. Because the branches in splitting the e-cash are much more than the K-tree, the amount of computation in generating the branches of the e-cash is increased too.

The e-cash in my paper simply saves $Q',\alpha_{(n)11,12,13} \cdot \beta_{(n)11,12,13} \cdot \gamma_{(n)11,12,13} \cdot p=112,12,13$. K (Assuming the K-tree assigns to the third branch) whose storage space is approximately $32+128+128+192+128+32=736$bit=92byte, which reduces at least 92% of the storage amount and the network bandwidth. The two-tree is used in [3]. Because the branches in splitting the e-cash are much more than the K-tree, the amount of computation in generating the branches of the e-cash is increased too.

The security mobile payment system [1] need bank to complete payment, while my paper needn’t in the micro-payment. Bank isn’t the bottle neck of e-cash.

The e-cash [4] cannot circulate in merchant’s commerce. The e-cash in my paper can circulate many times off-line before deposited to a bank.

V. CONCLUSION

Mobile multi-bank divisible e-cash protocol proposes on mobile terminals for electronic payments which can circulate in multiple banks and cannot limit from the bank that issues the e-cash. The divisible e-cash does not pass the bank by the consumption of each e-cash, thus the bank has not become the bottleneck of electronic cash transactions. The protocol of mobile electronic payment system is safe, simple, efficient, and suitable for the mobile payment terminals of which calculation capacity, storage, network bandwidth, power supply are very limited.

REFERENCES


Ting Huang was born in the city of Yichang, China on May 13th, 1979. Her degree of bachelor in communication engineering was earned from College of Communication Command in Wuhan, China in 2002 and her degree of master in computer was earned from China Three Gorges University in Yichang, China in 2010. Ting’s major field of study is information security and teaching of computer. She is the teacher of China Three Gorges University. Her published articles are “Study on Mobile Divisible E-cash Based on Elliptic Curves” (Journal of Wuhan university of technology, 2010),” Design of mobile electronic micro-payment system” (Applied mechanics and materials, 2014) and etc.