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 limit of 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 each 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 $a_0=K^{n-1}$, while the total amount of the entire classical K-ary tree is $a=K^{n-1}$ and the total amount of K-ary tree in this agreement is $a=nK^{n-1}$. The classical principle requires that the user withdraws money n times, the bank signs n 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 n 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 just1/n 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.

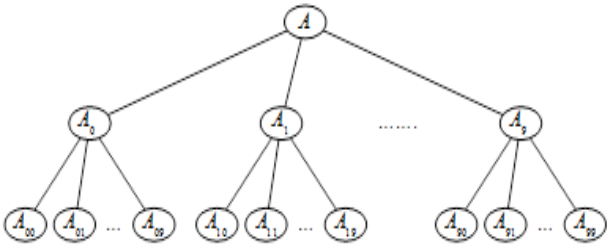

Fig. 1. Three layer of the 10-ary tree.

III. Design of Agreement

The point of abscissa and vertical axis are x and y on the elliptic curve respectively; SK_U, SK_C, SK_F, SK_M are the user, credit center, factory, merchant's private key; PK_U, PK_C, PK_F, PK_M are the user, credit center, factory, merchant's public key. The user's name is NA_U. Each bank is classified by a number, such as: CCB is the $bank_1$,…; another bank is the $bank_n$, of which the corresponding private key and the signature private key are defined as SK_{Bn}, SK_{Bn1}, SK_{Bn2} and the corresponding public key is PK_{Bn}, PK_{Bn1}, PK_{Bn2}, ($n^{\in(1,\infty)}$). H is the hash function.

TC is defined as the use cycle of the e-cash (TC is the time cycle T_1 to T_3). T_1 generates when the user gets the e-cash from the bank. The time cycle T_1 to T_2 is an effective use cycle TC' of the e-cash (TC' is the given value). When T_2 reaches the e-cash is unavailable. The user must go to the bank for updating the e-cash only before T_3 (The time cycle T_2 to T_3 is the updating cycle TC" of the e-cash that is the given value).

A. Withdrawal Agreement

Step 1: The user fetches Q yuan from the $bank_n$ and gets the segmentation parameter k of the e-cash. Then he gets $A\in_R Z_n$, $\rho=H(A+SK_U)$.Then he gets the k-ary tree by Q yuan, saves ρ, k to withdrawal database, and sends them to the $bank_n$ after encrypting ID_{nU}, ρ, Q,

Manuscript received January 18, 2015; revised March 26, 2015.

Ting Huang is with the College of Computer and Information Technology, China Three Gorges University, Yichang, Hubei, CO 443002, China (e-mail: 14863403@qq.com).

Step 2: The $bank_n$ gets the root node of the e-cash: $\alpha_{(n)} = H[(QSK_{Bn1} + \rho SK_{Bn2}) \| SK_{Bn} \| T_1 \| B]$, and computes $\beta_{(n)} = H[(CG)_x \| \alpha_{(n)} \| T_1 \| Q]$, $\gamma_{(n)} = C + \beta_{(n)} SK_{Bn}$, $B, C \in_R Z_n$. The $bank_n$ saves $\alpha_{(n)}$, ρ, k, T_1, B to the database , encrypts ID_{nU}, Q, $\alpha_{(n)}$, $\beta_{(n)}$, $\gamma_{(n)}$, T_1, and sends to the user.

Step 3: The user decrypts ID_{nU}, Q, $\alpha_{(n)}$, $\beta_{(n)}$, $\gamma_{(n)}$, T_1 and saves them to withdrawal database.

B. Payment Agreement

The user pays the e-cash to the $merchant_0$ through the credit center. The payments in two ways are described in Table I and Table II. The courses that $merchant_0$ pays the e-cash to Factory or $Merhant_i$ (i = 1, 2,…, n) are described in Table III and Table IV.

TABLE I: PAYMENT AGREEMENT (Q'=Q YUAN)

User	Credit center	Merchant
$D, E_1, E_2 \in Z_n, D \neq 0, q = H(\alpha_{(n)})$, $q_1 = DID_{nU} - qE_1$, $q_2 = D - qE_2$. Q, $\alpha_{(n)}$, $\beta_{(n)}$, $\gamma_{(n)}$, T_1 , q,q_1,q_2 , Timestamp are sent →	The credit center exams $\beta_{(n)} \stackrel{?}{=} H[(\gamma_{(n)} G - \beta_{(n)} PK_{Bn})_x \| \alpha_{(n)} \| T_1 \| Q]$ saves Q, $\alpha_{(n)}$, $\beta_{(n)}$, $\gamma_{(n)}$, q,q_1,q_2	
	It computers $G = H[Q \| (SK_C PK_M)_x]$ → It sends G, NA_U, Timestamp and notices merchant send goods	$G \stackrel{?}{=} H[Q \| (SK_M PK_C)_x]$
When he gets the goods, the user computes $F = H[(SK_U PK_C)_x]$, deletes $\alpha_{(n)}$, Q, sends F, NA_U, Timestamp, merchant's address →	It exams $F \stackrel{?}{=} H[(SK_C PK_U)_x]$. It computers $G' = H[Q \| (SK_C PK_M)_x \| \alpha_{(n)}]$, → sends Q, G', $\alpha_{(n)}$, NA_U, q,q_1,q_2, Timestamp	The merchant exams $G' \stackrel{?}{=} H[Q \| (SK_M PK_C)_x \| \alpha_{(n)}]$ and saves $\alpha_{(n)}$,Q, q,q_1,q_2

TABLE II: PAYMENT AGREEMENT (Q'<Q YUAN)

User	Credit center
$D, E_1, E_2 \in Z_n, D \neq 0$, $q = H(\alpha_{(n)})$, $q_1 = DID_{nU} - qE_1$, $q_2 = D - qE_2$,According to Q' the k-tree is generated: The user obtains $\alpha_{(n)l0} = \alpha_{(n)}$, for(l1=0;l1<=k;l1++){for(l2=0;l2<=k;l2++) {…{ for(ls=0;ls<=k;ls++){ $B_{l1,\ldots,ls}$,$C_{l1,\ldots,ls} \in Z_n$, $\alpha_{(n)l1,..ls} = H[B_{l1,\ldots,ls} \| T_1 \| SK_U \| \alpha_{(n)l1,..l(s-1)}]$ }}…}}, $\beta_{(n)l1,..ls} = H[(C_{l1,\ldots,ls} G)_x \| \alpha_{(n)l1,\ldots,ls} \| k \| \rho \| p[l1] \| ... \| p[ls] \| T_1 \| Q']$, $\gamma_{(n)l1,\ldots,ls} = C_{l1,\ldots,ls} + \beta_{(n)l1,\ldots,ls} SK_U$, p={ρ ,l1,…,ls}, and saves $B_{l1,\ldots,ls}$, $C_{l1,\ldots,ls}$ to the database. Q', p, $\alpha_{(n)l1,\ldots,ls}$, $\beta_{(n)l1,\ldots,ls}$, $\gamma_{(n)l1,\ldots,ls}$ $\alpha_{(n)}$, T_1 , q,q_1,q_2,Timestamp are sent. The unused e-cash can be used next time. →	The credit center exams $\beta_{(n)l1,\ldots,ls} \stackrel{?}{=} H[(\gamma_{(n)l1,\ldots,ls} G - \beta_{(n)l1,\ldots,ls} PK_U)_x \| \alpha_{(n)l1,\ldots,ls} \| k \| \rho \| p[l1] \| ... \| p[ls] \| T_1 \| Q']$, saves Q', p, $\alpha_{(n)l1,\ldots,ls}$, $\beta_{(n)l1,\ldots,ls}$, $\gamma_{(n)l1,\ldots,ls}$, q,q_1,q_2

User	Credit center	Merchant
	It computers $G = H[Q' \| (SK_C PK_M)_x]$ → and sends G, NA_U, Timestamp	$G \stackrel{?}{=} H[Q' \| (SK_M PK_C)_x]$ Send the goods to the user
The user gets the goods, computers $F = H[(SK_U PK_C)_x]$, deletes $\alpha_{(n)l1,\ldots,ls}$, $B_{l1,\ldots,ls}$, $C_{l1,\ldots,ls}$,Q', sends F, NA_U, Timestamp, →	It exams $F \stackrel{?}{=} H[(SK_C PK_U)_x]$, computers → $G' = H[Q' \| (SK_C PK_M)_x \| \alpha_{(n)kj}]$, sends Q', p, $\alpha_{(n)l1,\ldots,ls}$,NA_U, G', q, q_1, q_2, Timestamp	The merchant exams $G' \stackrel{?}{=} H[Q' \| (SK_M PK_C)_x \| \alpha_{(n)kj}]$ and save $\alpha_{(n)l1,\ldots,ls}$, p, Q',

TABLE III: MERCHANT'S PAYMENT AGREEMENT (Q'=Q YUAN)

$Merchant_0$	Credit center	$Merchant_i$
$D, E_1, E_2 \in Z_n, D \neq 0$, $q = H(\alpha_{(n)})$, $q_1 = DID_{nU} - qE_1$, $q_2 = D - qE_2$, → $H_{0(n)} = H[(wG)_x \| \alpha_{(n)} \| T_1 \| Q]$, $w \in Z_n$ $K_{0(n)} = H_{0(n)} S_M + w \cdot T_1$, q,q_1,q_2, Q, $\alpha_{(n)}$, Timestamp are sent	$H_{0(n)} \stackrel{?}{=} H[(K_{0(n)} G - H_{0(n)} P_M)_x \| \alpha_{(n)} \| T_1 \| Q]$. It saves Q, $\alpha_{(n)}$, q,q_1,q_2	
	It computers $G = H[Q \| (SK_C PK_{Mi})_x]$, → sends G, NA_U, Timestamp, and notices $merchant_i$ send goods.	$G \stackrel{?}{=} H[Q \| (SK_{Mi} PK_C)_x]$
After he gets the goods, the $merchant_0$ computes $F = H[(SK_{M0} PK_C)_x]$, deletes $\alpha_{(n)}$, Q. It sends Timestamp, F, NA_U, merchant's address →	$F \stackrel{?}{=} H[(SK_C PK_{M0})_x]$. It computers $G' = H[Q \| (SK_C PK_{M0})_x \| \alpha_{(n)}]$, → sends Q, G', $\alpha_{(n)}$, NA_U, q, q_1, q_2, Timestamp	The $merchant_i$ exams $G' \stackrel{?}{=} H[Q \| (SK_{M0} PK_C)_x \| \alpha_{(n)}]$ and saves $\alpha_{(n)}$, Q, q,q_1,q_2

TABLE IV: MERCHANT'S PAYMENT AGREEMENT (Q'<Q YUAN)

Merchant$_0$		Credit center
$D, E_1, E_2 \in Z_n, D \neq 0$, $q = H(\alpha_{(n)})$, $q_1 = DID_{nU} - qE_1$, $q_2 = D - qE_2$, According to Q' the k-tree is generated: The merchant$_0$ obtains $\alpha_{(n)l0} = \alpha_{(n)}$, for(l1=0;l1<=k;l1++) {for(l2=0;l2<=k;l2++){… { for(ls=0;ls<=k;ls++) { $B_{l1,\ldots,ls}, C_{l1,\ldots,ls} \in Z_n$, $\alpha_{(n)l1,..ls} = H[B_{l1,\ldots,ls} \| T_1 \| SK_{M0} \| \alpha_{(n)l1,..l(s-1)}]$ }}…}}, $H_{0(n)l1,\ldots,ls} = H[(C_{l1,\ldots,ls}G)_x \| \alpha_{(n)l1,\ldots,ls} \| k \| \rho \| p[l1] \| \ldots \| p[ls] \| T_1 \| Q']$, $K_{0(n)l1,\ldots,ls} = H_{0(n)l1,\ldots,ls}S_{M0} + C_{l1,\ldots,ls}$, p={ρ ,l1,…,ls}, and saves $B_{l1,\ldots,ls}, C_{l1,\ldots,ls}$ to the database. Q', p, $\alpha_{(n)l1,\ldots,ls}$, $H_{0(n)l1,\ldots,ls}$, $K_{0(n)l1,\ldots,ls}$, $\alpha_{(n)}$, T_1, q,q_1,q_2, Timestamp $H_{\alpha\ldots} \stackrel{?}{=} H[(K_{\alpha\ldots}G - H_{\alpha\ldots}P_{\alpha\ldots})_x \| \alpha_{\ldots} \| k \| \rho \| p[l1] \| \ldots \| p[ls] \| T_1 \| Q']$	→	The credit center exams

Merchant$_0$	Credit center	Merchant$_i$
	It computers $G = H[Q' \| (SK_C PK_{Mi})_x]$, sends G, NA_U, Timestamp. →	$\stackrel{?}{G} = H[Q' \| (SK_{Mi} PK_C)_x]$. Send the goods to the merchant$_0$
The merchant$_0$ gets the goods, computers $F = H[(SK_{M0} PK_C)_x]$, deletes $\alpha_{(n)l1,\ldots,ls}$, Q', $B_{l1,\ldots,ls}$, $C_{l1,\ldots,ls}$, sends F, NA_U, Timestamp, merchant's address →	It exams $\stackrel{?}{F} = H[(SK_C PK_{M0})_x]$, computers $G' = H[Q' \| (SK_C PK_{Mi})_x \| \alpha_{(n)l1,\ldots,ls}]$, → sends G', Q', $\alpha_{(n)l1,\ldots,ls}$, p, NA_U, q,q_1,q_2, Timestamp	The merchant$_i$ exams $\stackrel{?}{G'} = H[Q' \| (SK_{Mi} PK_C)_x \| \alpha_{(n)l1,\ldots,ls}]$, and save $\alpha_{(n)l1,\ldots,ls}$, Q', p, q,q_1,q_2

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$_n$, the e-cash will deposit to the bank$_n$; merchant deposits to the bank$_m$ (not the bank$_n$), the bank$_n$ will send depositing e-cash issued to the bank$_m$. After the e-cash in the bank$_m$ is dealt well the merchant finishes the depositing course. The central bank that has 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, and whether it is out of date in the database. If it has the same path p, you can conclude the reuse of the e-cash. Then the bank can find the dishonest user's identity who takes the e-cash through the withdrawal database. Otherwise, the bank accepts the deposit e-cash.

D. Update of the e-Cash

Step 1: The user regenerates $A \in_R Z_n$, $\rho = H(A+SK_U)$, 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_1: $\alpha_{(n)} = H[(QSK_{Bn1} + \rho SK_{Bn2}) \| SK_{Bn} \| T_1 \| B]$, computes $\beta_{(n)} = H[(CG)_x \| \alpha_{(n)} \| T_1 \| Q]$, $\gamma_{(n)} = C + \beta_{(n)} SK_{Bn}$, B, $C \in_R Z_n$, and saves $\alpha_{(n)}$, ρ, B, K,T_1 to the database. ID_{nU}, $\alpha_{(n)}, \beta_{(n)}, \gamma_{(n)}$, Q, T_1 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 of $\alpha_{(n)l1,\ldots,ls}$, $H_{0(n)l1,\ldots,ls}$, $K_{0(n)l1,\ldots,ls}$.

IV. ANALYSIS OF SECURITY AND EFFICIENCY

Based on blind signature protocol $\alpha_{(n)}, \beta_{(n)}, \gamma_{(n)}$ are the right signature of the e-cash Q.

Proof: If the validation of $\beta_{(n)} \stackrel{?}{=} H[(\gamma_{(n)} G - \beta_{(n)} PK_{Bn})_x \| \alpha_{(n)} \| T_1 \| Q]$ is established, that proofs $\gamma_{(n)} G - \beta_{(n)} PK_{Bn} = CG$. $\gamma_{(n)} = C + \beta_{(n)} SK_{Bn}$, so $\gamma_{(n)} G - \beta_{(n)} PK_{Bn} = (C + \beta_{(n)} SK_{Bn})G - \beta_{(n)} PK_{Bn} = CG$. Similarly we can prove that $\alpha_{(n)11,}$ …, $\alpha_{(n)11,\ldots,1s}$, $\beta_{(n)11,}$ …, $\beta_{(n)11,\ldots,1s}$, $\gamma_{(n)11,}$ …, $\gamma_{(n)11,\ldots,1s}$, $H_{0(n)l1,\ldots,ls}$, $K_{0(n)l1,\ldots,ls}$ are correct signature of the e-cash Q.

The e-cash and its signatures are unforgeable.

The e-cash and its signatures embed SK. Any attacker must get SK so that he can forge the signatures and the e-cash, which must solve ECDLP. ECDLP remains unsolvable, so the e-cash and its signatures are unforgeable.

When the e-cash is used normally, the identity of the user cannot be obtained by the e-cash. However, the bank will reveal the anonymity of the e-cash after the e-cash is spent repeatedly.

When he pays an e-cash, somebody will send the e-cash and q, q_1, q_2. If the person spends the e-cash repeatedly, the bank will find the thing when the e-cash is deposited. Because B is a signature factor of which the person signs e-cash during withdrawal that is not equal in different electronic cash. $q = H(\alpha_{(n)})$ and $\alpha_{(n)} = H[(QSK_{Bn1} + \rho SK_{Bn2}) \| SK_{Bn} \| T_1 \| B]$ are different when the purchaser's request is the same. The bank will get (q, q_1, q_2) and (q', q_1', q_2') from the person's different payment operations. $q_1' = DID_{nU} - q'E_1$, $q_2' = D - q'E_2$, $D \neq 0$, so ID_{nU}= (q'q_1-qq_1')/(q'q_2-qq_2')(modn)=(q'DID_{nU}-qDID_{nU})/(q'D-qD). The bank will be able to detect the person's identity ID_{nU}. So the protocol is safe and fair (Firstly the bank must be safe and reliable).

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 $C_1(2^{l_1}, S, T, \Phi, R, \tilde{V}_0, \tilde{V}_1, \tilde{V}_2, \tilde{V}_3)$($i$=3).The storage space is at least 1024*9=9216bit=1152byte.While paying a e-cash in my paper simply saves Q', $\alpha_{(n)11,12,13}$, $\beta_{(n)11,12,13}$, $\gamma_{(n)11,12,13}$, p={ρ,l1,l2,l3}, K (Assuming the K-tree assigns to the third branch) whose storage space is approximately 32+128+128+192+(128+32 × 3)+32bit=736bit=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 protocol more complicatedly. The bank that issues the e-cash may be cancelled in this paper, which just need announce the time after T and PK_{Bn} are illegal.

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

[1] H. Wei, "Design and realization of a security mobile payment system," M.S. thesis, Dept. Northwest Univ., Xi-an, China, 2012.

[2] Z. Eslami and M. Talebi, "A new untraceable off -line electronic cash system," *Electronic Commerce Research and Applications*, vol. 10, pp. 59-66, 2011.

[3] G. B. Hou, "Research on security efficient multi-bank divisible electronic cash protocol," Ph.D. dissertation, Dept. Yanshan Univ., Qin Huang Dao, China, 2012.

[4] Y. X. Wu, X. Wang, X. L. Zhang, X. G. Cheng, and H. Q. Xian, "A practical group blind signature scheme and its application in electronic cash system," *Journal of Qingdao University* (natural science edition), vol. 27, pp. 72-76, 2014.

[5] A. Menezes, T. Okamoto, and S. Vanstone, "Reducing elliptic curve logarithms to logarithms in a finite field," *IEEE Trans. on Information Theory*, vol. 5, pp. 1639-1646, 1993.

[6] X. W. Meng and B. H. Zhao, "Fairness-based multi-bank e-cash," *Computer Applications and Software*, vol. 28, no. 10, pp. 195-197, 2011.

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.