Multiple-Bank E-Cash without Random Oracles

Jiangxiao Zhang, Zhoujun Li, and Hua Guo

State Key Laboratory of Software Development Environment,
Beihang University, Beijing 100191, PRC
orange_0092008@163.com

Abstract. Multiple-bank e-cash (electronic cash) model allows users and merchants to open their accounts at different banks. Most e-cash systems in the literature have been proposed in the single bank model in which clients and merchants have accounts at the same bank. In recent years, some multiple-bank e-cash systems were proposed, but they were proven secure in the random oracle model. In this paper, based on the Groth-Sahai proof system and Ghadafi group blind signature, we construct a multiple-bank e-cash system which is proven secure in the standard model. We achieve the dual privacy requirement (i.e., the user anonymity and bank anonymity) by using the group blind signature. Our scheme can also trace the identity of the signer. At last, some security properties of our scheme, such as anonymity, unforgeability, identification of the double spender and exculpability, are proved in the standard model.

Keywords: multiple-bank e-cash, group blind signature, automorphic blind signature, Groth-Sahai proofs.

1 Introduction

Group blind signature and multiple-bank e-cash were firstly introduced by Lysyanskaya and Ramzan [1]. Group blind signature combines the properties of both a group signature scheme and a blind signature scheme and therefore it maintains the anonymity of the signer as well as the message to be signed. Multiple-bank e-cash system consists of a large group of banks which are monitored by the Central Bank, opener, users and merchants. Each bank can dispense e-cash. The users and merchants can open their accounts in different bank. To make e-cash systems acceptable to government, the multiple-bank e-cash system should provide the owner tracing, coin tracing, identification of the double spender and signer tracing.

The multiple-bank e-cash system consists of Group Manager (GM), opener (OP), multiple banks $B_1, \cdots, B_n$, the users $U_1, \cdots, U_n$ and the merchants $M_1, \cdots, M_n$. Existing multiple-bank e-cash systems were proven secure in the random oracle model. Some results [2,3] have shown that some schemes proven secure in the random oracle model, are not secure in the standard model. Therefore, we propose a multiple-bank e-cash system proven secure in the standard model. We consider the dual privacy requirement, such as the user anonymity and bank anonymity.

Related Work. Much research has been performed in the area of e-cash [4,5,6,7,8,9,10,11]. The compact e-cash scheme [4] allows a user to withdraw a wallet containing 2^L coins efficiently and satisfies all the security properties mentioned above. However, the number of the coins that the user wants must be chosen in the withdrawal protocol, and be spent one by one in the spending protocol. Belenkiy, Chase, Kohlweiss and Lysyanskaya [12] proposed a compact e-cash system with non-interactive spending in the standard model. This scheme is based on P-signature [13], simulatable verifiable random functions [14] and Groth-Sahai proofs systems [15]. Izabachene and Libert proposed the first divisible e-cash scheme [11] in the standard model. They used a different method to authenticate the spending path. Unfortunately, the communication complexity of the spending scheme is proportional to the level number of the spent node. Zhang *et al.* constructed an anonymous transferable conditional e-cash [16] in the standard model.

The concept of group blind signatures was first introduced by Lysyanskaya and Zulfikar [1], where it was mainly used to design a multiple bank e-cash system in which digital coins could be issued by different banks. Based on the group blind signature [1], Jeong and Lee constructed a multi-bank e-cash system [17]. However, the system is proven secure in the random oracle model. In 2008, a multiple-bank e-cash [18] is proposed by Wang et al. However, it does not satisfy the unforgeable requirement. In order to obtain unforgeability, Chen et al. proposed an e-cash system [19] with multiple-bank. All the security of above multiple-bank e-cash are proven in the random oracle model which is known not to accurately reflect world (see [2] for instance).

Using divertible zero-knowledge proofs [21], Nguyen et al. proposed a group blind signature [20]. However, to eliminate the interaction in proof, it relies on the Fiat-Shamir transformation. Therefore, it is proven secure in the random oracle model. In 2013, Ghadafi constructed a group blind signature [22] in the standard model. He gave the formalizing definitions of the group blind signature. He also offered the dual privacy requirement, such as the user anonymity and the signer anonymity.

Our Contribution. We construct a multiple-bank e-cash system by using automorphic blind signature scheme [23] and Ghadafi group blind signature [22] in the standard model. We make the following contribution.

– We can trace the identity of the signer by the opener.
– By introducing a security tag, we can recover the identity of double spender.
– We give the security proof in the standard model. The new system satisfies anonymity, unforgeability, identification of double spenders and exculpability.

Paper Outline. The rest of the paper is organized as follows. In Section 2 we present preliminaries on the various cryptographic tools and assumptions. Definitions and the security properties of divisible e-cash are presented in Section 3. In Section 4, we present our construction and give efficiency analysis. We give the security proof in Section 5. Finally we conclude in Section 6.

2 Preliminaries

2.1 Mathematical Definitions and Assumptions

Definition 1. ***(Pairing).*** *A pairing $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$ is a bilinear mapping from two group elements to a group element [15].*

- *$\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_3$ are cyclic groups of prime order p. The elements G, H generate $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively.*
- *$\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2$ is a non-degenerate bilinear map, so $\hat{e}(G, H)$ generates G_3 and for all $a, b \in \mathbb{Z}_n$ we have $\hat{e}(G^a, H^b) = \hat{e}(G, H)^{ab}$.*
- *We can efficiently compute group operations, compute the bilinear map and decide membership.*

Definition 2. ***(Diffie-Hellman Pair).*** *A pair $(x, y) \in \mathbb{G}_1 \times \mathbb{G}_2$ is defined as a $Diffie-Hellman$ pair [24], if there exists $a \leftarrow \mathbb{Z}_p$ such that $x = G^a, y = H^a$, where G, H generate $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively. We denote the set of $\mathcal{DH}$ pairs by $\mathcal{DH} = \{(G^a, H^a) | a \in \mathbb{Z}_p\}$.*

2.2 Mathematical Assumptions

The security of this scheme is based on the following existing mathematical assumptions, i.e., the Symmetric External Diffie-Hellman (SXDH) [15], AWF-CDH [26] and the asymmetric double hidden strong Diffie-Hellman assumption (q-ADH-SDH) [23].

Definition 3. ***(Symmetric External Diffie-Hellman).*** *Let $\mathbb{G}_1, \mathbb{G}_2$ be cyclic groups of prime order, G_1 and G_2 generate $\mathbb{G}_1$ and $\mathbb{G}_2$, and let $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$ be a bilinear map. The Symmetric External Diffie-Hellman (SXDH) Assumption states that the DDH problem is hard in both $\mathbb{G}_1$ and $\mathbb{G}_2$. For random a, b, $G_1, G_1^a, G_1^b \leftarrow \mathbb{G}_1$ and $G_2, G_2^a, G_2^b \leftarrow \mathbb{G}_2$ are given, it is hard to distinguish G_1^{ab} from a random element from $\mathbb{G}_i$ for $i = 1, 2$.*

Definition 4. ***(AWF-CDH).*** *Let $G_1 \leftarrow \mathbb{G}_1, G_2 \leftarrow \mathbb{G}_2$ and $a \leftarrow \mathbb{Z}_p$ be random. Given $(G_1, A = G_1^a, G_2)$, it is hard to output $(G_1^r, G_1^{ar}, G_2^r, G_2^{ar})$ with $r \neq 0$, i.e., a tuple (R, M, S, N) that satisfies*

$$\hat{e}(A, S) = \hat{e}(M, G_2) \qquad \hat{e}(M, G_2) = \hat{e}(G_1, N) \qquad \hat{e}R, G_2 = \hat{e}(G_1, S).$$

Definition 5. ***(q-ADH-SDH).*** *Let $G, F, K \leftarrow \mathbb{G}_1, H \leftarrow \mathbb{G}_2$ and $x, c_i, v_i \leftarrow Z_p$ be random. Given $(G, F, K, G^x; H, Y = H^x)$ and*

$$(A_i = (K \cdot G^{v_i})^{\frac{1}{x+c_i}}, B_i = F^{c_i}, D_i = H^{c_i}, V_i = G^{v_i}, W_i = H^{v_i})$$

for $1 \leq i \leq q-1$, it is hard to output a new tuple $(A = (K \cdot G^v)^{\frac{1}{x+c}}, B = F^c, D = H^c, V = G^v, W = H^v)$ with $(c, v) \neq (c_i, v_i)$ for all i. i.e., one that satisfies

$$\hat{e}(A, Y \cdot D) = \hat{e}(K \cdot V, H), \hat{e}(B, H) = \hat{e}(F, D), \hat{e}(V, H) = \hat{e}(G, W).$$

2.3 Useful Tools

Groth-Sahai Proofs. Groth and Sahai [15] constructed the first NIZK proof systems. They prove a large class of statements in the context of groups with bilinear map in the standard model. In order to prove the statement, the prover firstly commits to group elements. Then the prover produces the proofs and sends the commitments and the proofs to the verifier. And last the verifier verifies the correctness of the proof.

Groth-Sahai proofs can be instantiated under different security assumptions but since as noted by [25] the most efficient Groth-Sahai proofs are those instantiated under the SXDH assumption, we will be focusing on this instantiation. Following the definitions of Ghadafi [22], the proof system consists of the algorithms $GS = (GSSetup, GSProve, GSVerify, GSExtract, GSSimSetup,$ $GSSimProve)$. For ease of composition, we also define the algorithm $GSPOK$ $(crs, \{w_1, \ldots, w_i\}, \{\varepsilon_1 \wedge \ldots \wedge \varepsilon_j\})$ [22] which proves j multiple equations involving witness $(w_1, \ldots, w_i)$ and returns a vector of size j of Groth-Sahai proofs.

Automorphic Blind Signature. Abe et al. proposed an automorphic blind signature scheme [23] which is structure-preserving signatures on group elements. The automorphic blind signature to sign one message is done between U and signer. We define the automorphic blind signature to sign one message as $ABSign()$ and the verification of the signature as $ABSVerify()$.

In order to sign two messages, we use the definition 2 in [23] to finish a generic transformation [23] from any scheme signing two messages to one singing one message.

Group Blind Signature. Ghadafi constructed a group blind signature [22] which provides the dual privacy requirement. On the one hand, the signer (the bank) can hide his identity and parts of the signature that could identify him. On the other hand, the user can hide the message and parts of the signature which could lead to a linkage between a signature and its sign request. Ghadafi presented two example instantiations. We use the first construction and define the first construction as $GGBS()$. The CERT signature [22] is instantiated by using the above automorphic blind signature [23]. Note that when we sign one message, we use the $ABSign()$ to generate the signature. However, when we sign two messages, we use the transformation signature which transforms the signature to two messages into the signature to one message.

3 Definitions for Multiple-Bank E-Cash

Our model builds on the security models for the Ghadafi group blind signature [22] and the transferable e-cash [10]. The parties involved in a multiple-bank e-cash are: a group manager GM, an opener OP, many banks B_i and users U_i. Note that merchants M are the special users.

In the following, we first describe the algorithms for multiple-bank e-cash.

3.1 Algorithms

We represent a coin as *coin*, which its identity is *Id*. A multiple-bank e-cash system, denoted Π, is composed of the following procedures, where λ is a security parameter.

- ParamGen(1^λ) is run by some trusted third party (TTP) which takes as input 1^λ and outputs the public key *mbpk* for the multiple-bank e-cash system, the group manager's secret key sk_{GM} and the opener's extraction key (ck_{op}, ek_{op}).
- BKeyGen() is run by the banks B_i, to generate his pairs of personal secret/pulbic keys $(bssk_i, bspk_i)$ and $(bgsk_i, bgpk_i)$. The former is used in the joining protocol. The latter is used for issuing the group blind signature in the withdrawal protocol. We assume that the public key is publicly accessible.
- UKeyGen() is run by the users U_i, to generate his pair of personal secret/pulbic keys (sk_{U_i}, pk_{U_i}). Note that the merchants M are special users.
- Issue($B_i(mbpk, i, bssk_i)$,$GM(sk_{GM}, i, bspk_i)$) is an interactive protocol between a bank B_i and the group manager GM. After a successful completion of this protocol, B_i becomes a member of the group. If successful, B_i obtains the certificate $cert_i$, and stores the second secret/public keys $(bgsk_i, bgpk_i)$ and $cert_i$ into gsk_i. GM obtains the signature sig_i on the second public key $bgpk_i$ and stores the second public key $bgpk_i$ and sig_i into reg_i.
- Withdraw($U_i(mbpk, m)$,$B_i(gsk_i, pk_{U_i})$) is an interactive protocol between a user U_i and an anonymous bank B_i. If the protocol completes successfully, U_i obtains a blind signature π_m on the message m. B_i does not learn what the message was. U_i only knows the signature that is signed by the bank, but he does not know which bank issues the signature.
- Spend($U_i(coin, pk_M, sk_{U_i}, pk_{U_i}, ck_{op})$,$M(sk_M, pk_M, bgpk_i)$) is an interactive protocol between a user U_i and a merchant M. If the protocol completes successfully, U_i obtains the corresponding serves. M obtains an e-coin *coin*.
- Deposit($M(coin, sk_M, pk_M, bgpk_i)$,$B_i(pk_M, DB)$) is an interactive protocol between a merchant M and a bank B_i. If *coin* is not valid, B_i outputs $\perp$. Else, B_i checks whether the database DB contains an e-cash $coin'$ in which the serial number is the same as the one in *coin*. If DB contains $coin'$, B_i outputs $(coin, coin')$. Else, B_i adds *coin* to DB, and credits M's account.
- Identify($coin, coin'$) is a deterministic algorithm executed by B_i. It outputs the public key pk_{U_i} and a proof τ_G.
- VerifyGuilt(pk_{U_i}, τ_G) is a deterministic algorithm that can be executed by anyone. It outputs 1 if τ_G is correct and 0 otherwise.
- Open($mbpk, ek_{op}, reg_i, m, \pi_m$) is a deterministic algorithm in which the opener uses his extraction key ek_{op} to recover the identity i of the banker and produces a proof τ_S attesting to this claim.
- VerifySigner($mbpk, i, bspk_i, m, \pi_m, \tau_S$) is a deterministic algorithm which inputs an index i and returns 1 if the signature π_m was produced by the bank B_i or 0 otherwise.

3.2 Security Properties

In this section, we give the brief description of the security properties in our scheme.

Anonymity. Anonymity includes the bank anonymity and the user anonymity. In the following, we give the formal definition of the bank anonymity and the user anonymity.

The bank anonymity guarantees that the adversary is unable to tell which bank produced a signature. We employ the same signer anonymity used by [22], where we require that the adversary is given two banks of its choice, the adversary still cannot distinguish which of the two banks produced a signature.

The user anonymity guarantees that the adversary is unable to tell which message it is signing. We employ the blindness property used by [22], where we require that the adversary is given a signature on a message m_i for $i = \{0, 1\}$ of its choice, he still cannot distinguish which of the two messages is signed.

Unforgeability. Unforgeability guarantees that no collection of users can ever spend more coins than they withdrew. Formally, we have the following definition based on the experiment given below.

Identification of Double-Spender. The identification of double-spender guarantees that no collection of users, collaborating with the merchant, can spend an e-cash twice without revealing one of their identities. Formally, we have the following experiment.

Exculpability. The exculpability guarantees that the bank, even when colluding with malicious users, cannot falsely accuse hones users of having double-spent a coin. Formally, we have the following experiment.

4 Multiple-Bank E-Cash

Multiple-bank e-cash allows users and merchants to open their accounts at different banks. It supplies the users anonymity and the banks anonymity. In the following, we give the details of this scheme.

4.1 Setup

On input 1^λ and output the public parameters of bilinear groups $bgpp = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3, \hat{e}, G, H)$, where λ is the security parameter. We choose random elements $F, K, T \in G_1$. On input $bgpp$ run the setup algorithms for Groth-Sahai proofs and return two reference strings crs_1, crs_2 and the corresponding extraction keys ek_1, ek_2. The two reference strings and the extraction keys are used for the first round and the second round of the automorphic blind signature scheme which is used for issuing the group blind signature on the e-cash. The opener's commitment key and extraction key are $(ck_{op} = crs_2, ek_{op} = ek_2)$. On choose $s_{GM} \leftarrow \mathbb{Z}_p$ and output the key pair of group manager $(sk_{GM} = s_{GM}, pk_{GM} = (S_1 = G_1^{s_{GM}}, S_2 = G_2^{s_{GM}}))$. The public key of multiple-bank e-cash is $mbpk = (bgpp, F, K, T, crs_1, crs_2, pk_{GM})$. $\mathcal{H}$ is a collision-resistant hash function.

Each bank B_i chooses $s_{B_i}, s'_{B_i} \leftarrow \mathbb{Z}_p$ and creates two key pairs $(bssk_{B_i} = s_{B_i}, bspk_{B_i} = (S_1 = G_1^{s_{B_i}}, S_2 = G_2^{s_{B_i}}))$ and $(bgsk_{B_i} = s'_{B_i}, bgpk_{B_i} = (S_1 = G_1^{s'_{B_i}}, S_2 = G_2^{s'_{B_i}}))$. The first key pair is the bank's personal key pair. The second one is used for signing the e-cash. Each user U_i chooses $s_{U_i} \leftarrow \mathbb{Z}_p$ and generates the key pair $(sk_{U_i} = s_{U_i}, pk_{U_i} = (S_1 = G_1^{s_{U_i}}, S_2 = G_2^{s_{U_i}}))$. Each merchant M_i also creates the key pair $(sk_{M_i} = s_{M_i}, pk_{M_i} = (S_1 = G_1^{s_{M_i}}, S_2 = G_2^{s_{M_i}}))$.

4.2 The Joining Protocol

The joining protocol allows the bank to obtain a certificate from the group manager. In order to issuing e-cash, each bank firstly joins into the group whose manager is GM. Then the bank B_i obtains the certificate $cert_i$. Using the certificate and the key pair $(bgsk, bgpk)$, the bank issuing the e-cash. In the following, we give the details of the protocol.

1. $(B_i \rightarrow GM)$. The bank B_i generates the signature $sig_i = ABSign(bssk_i, bgpk_i)$. Then B_i sends $sig_i, bgpk_i = (S_1^{bg} = G_1^{s'_{B_i}}, S_2^{bg} = G_2^{s'_{B_i}})$ to the group manager GM.
2. $(GM \rightarrow B_i)$. GM checks whether the public key $bgpk_i$ has existed in the database DB_{pk} or verifies $\hat{e}(S_1^{bg}, G_2) \neq \hat{e}(G_1, S_2^{bg})$. If it is not, GM verifies the signature sig_i. If $ABSverify(bspk_i, bgpk_i, sig_i) = 1$, GM generates the certificate $cert_i = ABSign(sk_{GM}, bgpk_i)$. At last, GM sends $cert_i$ to B_i.
3. B_i verifies the correctness of the certificate. If $ABSverify(pk_{GM}, bgpk_i, cert_i) = 1$, B_i saves the certificate $cert_i$.

4.3 The Withdrawal Protocol

The withdrawal protocol allows U_i to withdraw an e-cash from B_i. c_m is defined as the commitment and corresponding proof to the message m. $C_m^{ck_{op}}$ is defined as the commitment to m using the commitment key ck_{op}. In the following, we give the protocol in detail.

1. $(U_i \rightarrow B_i)$. U_i chooses $s_m \leftarrow \mathbb{Z}_p$ and generates the serial number $S = (G_1^{s_m}, G_2^{s_m})$. U_i also chooses $q_1, q_2 \leftarrow \mathbb{Z}_p$ and computes $Q_1 = G_1^{q_1}, Q_2 = G_2^{q_1}, Q_3 = G_1^{q_2}, Q_4 = G_1^{q_2}$. U_i picks at random nonces $\iota_1, \iota_2 \leftarrow \mathbb{Z}_p$. To hide the serial number, U_i generates the following commitments c_S and the correct proofs π_S by using the commitment key ck_{op} of the opener. U_i also generates

the commitment $c_{pk_{U_i}}$ and the correct proof $\pi_{pk_{U_i}}$ to U_i's public key $pk_{U_i} = (S_1 = G_1^{s_{U_i}}, S_2 = G_2^{s_{U_i}})$.

$$\begin{aligned}
c_S &= (C_{G_1^{s_m}}^{ck_{op}}, C_{G_2^{s_m}}^{ck_{op}}, C_{Q_1}^{ck_{op}}, C_{Q_2}^{ck_{op}}, U_1 = T^{\iota_1} \cdot G_1^{s_m}), \\
\pi_S &\leftarrow GSPOK\{crs_1, \{G_1^{s_m}, G_2^{s_m}, Q_1, Q_2\}, \hat{e}(G_1^{s_m}, G_2) = \hat{e}(G_1, G_2^{s_m}) \wedge \\
&\qquad \hat{e}(Q_1, G_2) = \hat{e}(G_1, Q_2) \wedge \hat{e}(T, Q_2) \cdot \hat{e}(G_1^{s_m}, G_2) = \hat{e}(U_1, G_2)\}, \\
c_{pk_{U_i}} &= (C_{G_1^{s_{U_i}}}^{ck_{op}}, C_{G_2^{s_{U_i}}}^{ck_{op}}, C_{Q_3}^{ck_{op}}, C_{Q_4}^{ck_{op}}, U_2 = T^{\iota_2} \cdot G_1^{s_{U_i}}), \\
\pi_{pk_{U_i}} &\leftarrow GSPOK\{crs_1, \{G_1^{s_{U_i}}, G_2^{s_{U_i}}, Q_3, Q_4\}, \hat{e}(G_1^{s_{U_i}}, G_2) = \hat{e}(G_1, G_2^{s_{U_i}}) \wedge \\
&\qquad \hat{e}(Q_1, G_2) = \hat{e}(G_1, Q_2) \wedge \hat{e}(T, Q_2) \cdot \hat{e}(G_1^{s_{U_i}}, G_2) = \hat{e}(U_2, G_2)\}.
\end{aligned}$$

At last, U_i sends $\{pk_{U_i}, c_S, \pi_S, c_{pk_{U_i}}, \pi_{pk_{U_i}}\}$ to B_i.

2. $(B_i \to U_i)$. B_i verifies the public key pk_{U_i}, π_S and $\pi_{pk_{U_i}}$. If $GSVerify(crs_1, \pi_S) = 1$ and $GSVerify(crs_1, \pi pk_{U_i}) = 1$, B_i generates the group blind signature $GGBS(S, pk_{U_i})$ [22] on c_S and $c_{pk_{U_i}}$ by using the definition 2 in [23]. $GGBS(S, pk_{U_i})$ includes the signature $\sigma_{(S,pk_{U_i})}$ and the proof $\pi_{\sigma_{(S,pk_{U_i})}}$. $\pi_{\sigma_{(S,pk_{U_i})}}$ gives a proof that $\sigma_{(S,pk_{U_i})}$ is a valid signature on c_S and $c_{pk_{U_i}}$. At last, B_i sends $\pi_{\sigma_{(S,pk_{U_i})}}$ to U_i.
3. U_i verifies $\pi_{\sigma_{(S,pk_{U_i})}}$. If it is OK, to obtain the signature of the messages $(G_1^{s_m}, G_2^{s_m})$ and $(G_1^{s_{U_i}}, G_2^{s_{U_i}})$, U_i re-randomizes the proof $\pi_{\sigma_{(S,pk_{U_i})}}$ into $\pi'_{\sigma_{(S,pk_{U_i})}}$.
At last, U_i obtains the e-cash $coin = \{S, c_{pk_{U_i}}, \pi'_{\sigma_{(S,pk_{U_i})}}\}$.

4.4 The Spending Protocol

The spending protocol allows U_i to spend an e-cash to the merchant M. In order to hide U_i's identity, U_i randomizes $c_{pk_{U_i}}$ and $\pi'_{\sigma_{pk_{U_i}}}$ by $RdCom$ and $RdProve$.

1. $(M \to U_i)$. M computes $R = \mathcal{H}(pk_M||Date||r)$ and sends $\{R, pk_M, Date, r\}$ to U_i, where $r \leftarrow \mathbb{Z}_p$ is a random value.
2. $(U_i \to M)$. U_i also computes $R = \mathcal{H}(pk_M||Date||r)$. The commitment to U_i's public key is $c_{pk_{U_i}} = (C_{G_1^{s_{U_i}}}^{ck_{op}}, C_{G_2^{s_{U_i}}}^{ck_{op}}, C_{Q_3}^{ck_{op}}, C_{Q_4}^{ck_{op}}, U_1 = T^{\iota_2} \cdot G_1^{s_{U_i}})$. In order to hide U_i's public key, U_i chooses $\iota_2', t', \mu', \nu', \rho' \leftarrow \mathbb{Z}_p$ and randomizes $c_{pk_{U_i}}$ into $c'_{pk_{U_i}}$ [24].
U_i computes $Y = G_1^{\frac{1}{s_m+1}}$ and the security tag $T = pk_{U_i} \cdot \hat{e}(Y, G_2^R)$. Meanwhile, U_i gives the following NIZK proofs π_Y, π_T. π_Y gives a proof that $Y = G_1^{\frac{1}{s_m+1}}$ and s_m in Y is equal to s_m in S. π_T gives a proof that the security tag T is correctly formed.

$$\begin{aligned}
&\pi_Y \leftarrow GSPOK\{crs_1, \{Y, \phi_Y, s_m\}, \hat{e}(G_1^{\frac{1}{s_m+1}}, G_2^{s_m} \cdot G_2) = 1_{\mathbb{G}_3}, \hat{e}(Y/\phi_Y, h^\theta) = 1_{\mathbb{G}_3}\}, \\
&\pi_T \leftarrow GSPOK\{crs_1, \{T, \phi_T, Y, \phi_Y\}, \{T = pk_{U_i} \cdot \hat{e}(Y, G_2^R), \hat{e}(Y/\phi_Y, h^\theta) = 1_{\mathbb{G}_3}, \\
&\qquad \hat{e}(T/\phi_T, h^\theta) = 1_{\mathbb{G}_3}\}\},
\end{aligned}$$

where $\phi_Y, \phi_T \leftarrow \mathbb{G}_1$ are auxiliary variables, and $\theta \leftarrow \mathbb{Z}_p$ is a variable [11]. At last, U_i sends the e-cash $coin' = \{S, R, c'_{pk_{U_i}}, T, \pi_{coin'} = \{\pi'_{\sigma(S, pk_{U_i})}, \pi_Y, \pi_T\}\}$ to M.

3. M verifies the proofs. If they are correct, M saves $coin'$ and supplies serves to U_i.

4.5 The Deposit Protocol

M can deposit the e-cash to any bank. We assume that M has an account in B_i. When M wants to deposit a coin $coin'$ to B_i, M just sends $coin'$ to B_i. B_i checks the validity of $\pi_{coin'}$ and the consistency with S. If $coin'$ is not a valid coin, B_i rejects the deposit. Else, B_i checks if there is already the serial number S in the database. If there is not entry in the database, then B_i accepts the deposit of the coin $coin'$, credits the M's account and adds $coin'$ in the database. Else, there is an entry $coin'' = \{S, R', c''_{pk_{U_i}}, T', \pi'_{coin'}\}$ in the database. Then, B_i checks the freshness of R in $coin'$ compared to $coin''$. If it is not fresh, M is a cheat and B_i refused the deposit. If R is fresh, B_i accepts the deposit of the $coin'$, credits the M's account and add $(coin', coin'')$ to the list of double spenders. For recovering the identity of double spender, B_i executes the **Identify** algorithm.

4.6 Identify

The Identify algorithm makes sure that when a double-spending is found, B_i recovers the identity of double spender. The description of the Identify algorithm is as follow.

B_i knows two coins $coin_1 = \{S, R_1, \{c'_{pk_{U_i}}\}_1, T_1, \pi_1\}$ and $coin_2 = \{S, R_2, \{c'_{pk_{U_i}}\}_2, T_2, \pi_2\}$. Therefore, B_i directly recovers the public key pk_{U_i} by computing $(T_1^{R_2}/T_2^{R_1})^{\frac{1}{R_2 - R_1}}$.

4.7 Verify

Any one can verify the correctness of the double spenders and the signer (bank). In order to verify the correctness of the double spenders, any one executes the algorithm $VerifyGuilt$. One can parse the $coin_1$ and $coin_2$ as $(S, R_1, \{c'_{pk_{U_i}}\}_1, T_1, \pi_1)$ and $(S, R_2, \{c'_{pk_{U_i}}\}_2, T_2, \pi_2)$ and next run Identify on these values. If the algorithm Identify returns a public key, then one can check if π_1 is consistent with $(S, R_1, \{c'_{pk_{U_i}}\}_1, T_1)$ and if π_2 is consistent with $(S, R_2, \{c'_{pk_{U_i}}\}_2, T_2)$.

In order to verify the correctness of the signer (bank) who is opened by opener, any one executes the algorithm $VerifySigner$. The input of the algorithm is $(mbpk, i, bspk_i, m, \pi_m, \tau_S)$. After verifying the correctness of π_m, any one can check if the signature is signed by the bank.

4.8 Signer Tracing

Signer tracing is that the opener OP can recover the identity of the signer B_i. U_i obtains the signature $\pi_{(\sigma_S,\sigma_{pk_{U_i}})}$. Using the Open algorithm [22], OP extracts $(\sigma_{(S,pk_{U_i})}, cert_i, bgpk_i)$ from $\pi_{(\sigma_S,\sigma_{pk_{U_i}})}$. Therefore, we know which bank signs the e-cash.

4.9 Efficiency Analysis

We analyze the efficiency of our scheme, Chen's scheme [19] and Jeong's scheme [17] from the following 5 aspects, namely the efficiency of the joining protocol, the efficiency of the withdrawal protocol, the efficiency of the spending protocol, the efficiency of the deposit protocol and security model. It is somehow hard to quantify the exact cost of the spending protocol in [17] as the instantiation of the SKREP is very complex. We thus simplify the comparison by stating the total multi-exponentiations needed. The comparison is given in the following Table 1.

We assume that C_1 is the computation cost of the joining protocol. C_2 is the efficiency of the withdrawal protocol. C_3 is the efficiency of the spending protocol. C_4 is the efficiency of the deposit protocol. C_5 is the security model. ME represents the number of multi-exponentiation. ROM represents the random oracle model. SM represents the standard model.

Table 1. Efficiency comparison between related work and our scheme

	C_1		C_2		C_3		C_4		C_5
Chen's scheme [19]	B_i	$62ME$	B_i	$91ME$	M	$580ME$	B_i	$285ME$	ROM
	GM	$273ME$	U_i	$71ME$	U_i	$61ME$			
Jeong's scheme [17]	B_i	$11ME$	B_i	$10ME$	M	$12ME$	B_i	$12ME$	ROM
	GM	$12ME$	U_i	$19ME$	U_i	$11ME$			
Ours	B_i	$17ME$	B_i	$17ME$	M	$101ME$	B_i	$76ME$	SM
	GM	$17ME$	U_i	$36ME$	U_i	$26ME$			

Based on the Table 1, the number of multi-exponentiation in our scheme is less than one in Chen's scheme [19], but more than Jeong's scheme [17]. However, our scheme is proven secure in the standard model. We know that the scheme proven secure in the standard model is more securer than one proven secure in the random oracle model. Therefore, our scheme is more secure.

5 Security Analysis

Regarding the security of our construction, We have the following theorem.

Theorem 1. *Our multiple-bank e-cash system is secure under the following assumptions: unforgeability of automorphic blind signature and Ghadafi group blind signature, pseudorandomness of the Dodis-Yampolskiy PRF and soundness, witness indistinguishability and re-randomness of Groth-Sahai proofs system.*

6 Conclusion

In this paper, we present a multiple-bank e-cash which is proved secure in the standard model. We achieve the dual privacy requirement (the users anonymity and the bank anonymity) by using the Ghadafi group blind signature. To hide the identity of the user, we re-randomize the commitment to the user's public key by using the re-randomness of the Groth-Sahai proofs system. To ensure the security of the security tag, we use the pseudorandomness of the Dodis-Yampolskiy PRF. At last, we prove the security properties in the standard model.

Acknowledgments. This work was supported by the National Natural Science Foundation of China (grant number 60973105, 90718017, 61170189, 61300172), the Research Fund for the Doctoral Program of Higher Education (grant number 20111102130003, 20121102120017) and the Fund of the State Key Laboratory of Software Development Environment (grant number SKLSDE-2011ZX-03, SKLSDE-2012ZX-11).

References

1. Lysyanskaya, A., Ramzan, Z.: Group blind signature: a scalable solution to electronic cash. In: Hirschfeld, R. (ed.) FC 1998. LNCS, vol. 1465, pp. 184–197. Springer, Heidelberg (1998)
2. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited. In: 30th AcM STOC, pp. 209–218. ACM Press (1998)
3. Bellare, M., Boldyreva, A., Palacio, A.: An uninstantiable random-oracle-model scheme for a hybrid-encryption problem. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 171–188. Springer, Heidelberg (2004)
4. Camenisch, J.L., Hohenberger, S., Lysyanskaya, A.: Compact E-cash. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 302–321. Springer, Heidelberg (2005)
5. Canard, S., Gouget, A.: Divisible e-cash systems can be truly anonymous. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 482–497. Springer, Heidelberg (2007)
6. Au, M.H., Wu, Q., Susilo, W., Mu, Y.: Compact e-cash from bounded accumulator. In: Abe, M. (ed.) CT-RSA 2007. LNCS, vol. 4377, pp. 178–195. Springer, Heidelberg (2006)
7. Au, M.H., Susilo, W., Mu, Y.: Practical anonymous divisible e-cash from bounded accumulators. In: Tsudik, G. (ed.) FC 2008. LNCS, vol. 5143, pp. 287–301. Springer, Heidelberg (2008)
8. Belenkiy, M., Chase, M., Kohlweiss, M., Lysyanskaya, A.: Compact E-cash and simulatable VRFs revisited. In: Shacham, H., Waters, B. (eds.) Pairing 2009. LNCS, vol. 5671, pp. 114–131. Springer, Heidelberg (2009)
9. Canard, S., Gouget, A.: Multiple Denominations in E-cash with Compact Transaction Data. In: Sion, R. (ed.) FC 2010. LNCS, vol. 6052, pp. 82–97. Springer, Heidelberg (2010)
10. Blazy, O., Canard, S., Fuchsbauer, G., Gouget, A., Sibert, H., Traoré, J.: Achieving optimal anonymity in transferable e-cash with a judge. In: Nitaj, A., Pointcheval, D. (eds.) AFRICACRYPT 2011. LNCS, vol. 6737, pp. 206–223. Springer, Heidelberg (2011)

11. Izabachène, M., Libert, B.: Divisible e-cash in the standard model. In: Abdalla, M., Lange, T. (eds.) Pairing 2012. LNCS, vol. 7708, pp. 314–332. Springer, Heidelberg (2013)
12. Belenkiy, M., Chase, M., Kohlweiss, M., Lysyanskaya, A.: Compact E-Cash and Simulatable VRFs Revisited. In: Shacham, H., Waters, B. (eds.) Pairing 2009. LNCS, vol. 5671, pp. 114–131. Springer, Heidelberg (2009)
13. Belenkiy, M., Chase, M., Kohlweiss, M., Lysyanskaya, A.: P-signatures and noninteractive anonymous credentials. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 356–374. Springer, Heidelberg (2008)
14. Chase, M., Lysyanskaya, A.: Simulatable vrfs with applications to multi-theorem nizk. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 303–322. Springer, Heidelberg (2007)
15. Goyal, V., Mohassel, P., Smith, A.: Efficient non-interactive proof systems for bilinear groups. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008)
16. Zhang, J., Li, Z., Guo, H.: Anonymous transferable conditional E-cash. In: Keromytis, A.D., Di Pietro, R. (eds.) SecureComm 2012. LNICST, vol. 106, pp. 45–60. Springer, Heidelberg (2013)
17. Jeong, I.R., Lee, D.-H.: Anonymity control in multi-bank E-cash system. In: Roy, B., Okamoto, E. (eds.) INDOCRYPT 2000. LNCS, vol. 1977, pp. 104–116. Springer, Heidelberg (2000)
18. Wang, S., Chen, Z., Wang, X.: A new certificateless electronic cash scheme with multiple banks based on group signatures. In: Proc. of 2008 International Symposium on Electronic Commerce and Security, pp. 362–366 (2008)
19. Chen, M., Fan, C., Juang, W., Yeh, Y.: An efficient electronic cash scheme with multiple banks using group signature. International Journal of Innovative Computing, Information and Control 8(7) (2012)
20. Nguyen, K.Q., Mu, Y., Varadharajan, V.: Divertible zero-knowledge proof of polynomial relations and blind group signature. In: Pieprzyk, J.P., Safavi-Naini, R., Seberry, J. (eds.) ACISP 1999. LNCS, vol. 1587, pp. 117–128. Springer, Heidelberg (1999)
21. Okamoto, T., Ohta, K.: Divertible zero knowledge interactive proofs and commutative random self-reducibility. In: Quisquater, J.-J., Vandewalle, J. (eds.) EUROCRYPT 1989. LNCS, vol. 434, pp. 134–149. Springer, Heidelberg (1990)
22. Ghadafi, E.: Formalizing group blind signatures and practical constructions without random oracles. In: Boyd, C., Simpson, L. (eds.) ACISP. LNCS, vol. 7959, pp. 330–346. Springer, Heidelberg (2013)
23. Abe, M., Fuchsbauer, G., Groth, J., Haralambiev, K., Ohkubo, M.: Structure-preserving signatures and commitments to group elements. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 209–236. Springer, Heidelberg (2010)
24. Fuchsbauer, G.: Commuting signatures and verifiable encryption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 224–245. Springer, Heidelberg (2011)
25. Ghadafi, E., Smart, N.P., Warinschi, B.: Groth-Sahai proofs revisited. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 177–192. Springer, Heidelberg (2010)
26. Fuchsbauer, G.: Automorphic Signatures in Bilinear Groups and an Application to Round-Optimal Blind Signatures. In: Cryptology ePrint Archive, Report 2009/320, `http://eprint.iacr.org/2009/320.pdf`
27. Dodis, Y., Yampolskiy, A.: A Verifiable Random Function with Short Proofs and Keys. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 416–431. Springer, Heidelberg (2005)