Compact E-Cash with Practical and Complete Tracing

Bin Lian[1,2*], Gongliang Chen[2], Jialin Cui[1] and Dake He[3]
[1] Ningbo Institute of Technology, Zhejiang University, Ningbo, 315100 - China
[2] School of Information Security Engineering, Shanghai Jiao Tong University, Shanghai, 200240 - China
[3] School of information science and technology, Southwest Jiaotong University, Chengdu, 610031 - China
[e-mail: lianbin_a@163.com; chengl@sjtu.edu.cn; cuijl_jx@163.com; dkhe_scce@home.swjtu.edu.cn]
*Corresponding author: Bin Lian

Received August 14, 2018; revised October 18, 2018; accepted January 20, 2019; published July 31, 2019

Abstract

E-cash has its merits comparing with other payment modes. However, there are two problems, which are how to achieve practical/complete tracing and how to achieve it in compact E-cash. First, the bank and the TTP (i.e., trusted third party) have different duties and powers in the reality. Therefore, double-spending tracing is bank's task, while unconditional tracing is TTP's task. In addition, it is desirable to provide lost-coin tracing before they are spent by anyone else. Second, compact E-cash is an efficient scheme, but tracing the coins from double-spender without TTP results in poor efficiency. To solve the problems, we present a compact E-cash scheme. For this purpose, we design an embedded structure of knowledge proof based on a new pseudorandom function and improve the computation complexity from $O(k)$ to $O(1)$. Double-spending tracing needs leaking dishonest users' secret knowledge, but preserving the anonymity of honest users needs zero-knowledge property, and our special knowledge proof achieves it with complete proofs. Moreover, the design is also useful for other applications, where both keeping zero-knowledge and leaking information are necessary.

Keywords: Compact E-cash, practical tracing, complete tracing, special knowledge proof, knowledge-leak, zero-knowledge proof

This research work is supported by Zhejiang Provincial Natural Science Foundation of China (No: LY17F020019, LY16F010019), National Natural Science Foundation of China (No: 61271220), National key research and development plan (No: 2017YFB0802505) and Natural Science Foundation of Ningbo (No: 2016A610212).

http://doi.org/10.3837/tiis.2019.07.022 **ISSN : 1976-7277**

1. Introduction

Nowadays, e-Commerce system [1, 2] is common, and e-payment is the core module. Some e-payments guarantee the anonymity of users, but sometimes it is abused for crimes. Over the past years, quite some research effort has been put in design of E-cash [3] based on blind signature [4, 5] or knowledge signature [6, 7]. To prevent user from abusing anonymity, the tracing function is necessary. **Fig. 1** shows a typical E-cash model.

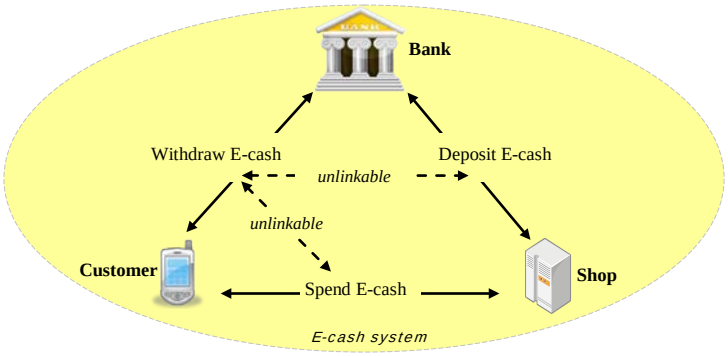


Fig. 1. Typical E-cash model

In general, an E-cash system has the following properties:

- **Anonymity**: No one can identify the spender or link the spending behaviors.
- **Unreusability**: The owner of e-cash will be identified if he spends one coin twice.
- **Unforgeability**: No one except the bank can generate valid e-cash.
- **Offline mode**: There is no the third party participating in the payment process.
- **Environmental independence**: System security depends on the cryptographic scheme.

Some interesting E-cash schemes such as divisible E-cash, transferable E-cash and changeable denomination E-cash, etc., are put forward. And compact E-cash [8] is the important scheme. In such system, user withdraws 2^l e-coins by performing the withdrawal protocol one time and stores the coins in $O(l)$ bits.

2. Related Work

Our goal is to construct a compact E-cash system with practical and complete tracing, so we present two parts of the related work.

Part I: the Practical tracing and the Complete tracing

"Practical" means that different tracings should be provided by the appropriate entity, and "Complete" means that the tracings should meet the demand from real-world applications.

To prevent customer from reusing a coin, double-spending tracing is the basic function. Since it is related to the bank's business, double-spending tracing should be performed by bank.

Sometimes, the anonymity of users may provide the convenience for crimes, such as money laundering and blackmailing. Therefore, S. von Solms and D. Naccache [9] suggested that the anonymity of users should be revoked when necessary, and then [10] and [11] put forward a new E-cash scheme, i.e., fair E-cash. In such a system, the user remains anonymous if he honestly spends the legal e-coins, but if crimes occur so that the related transaction is illegal, the e-coins and the owner can be traced unconditionally. Here "unconditionally" means that the coins and the owner can be traced when the user does not double-spend coins.

In reality, the authorities are usually the trusted third parties, who have the power to perform unconditional tracing only when the E-cash is involved in crimes.

Therefore, it is not practical that the TTP is indispensable to double-spending tracing [1, 10, 12, 13, 14, 15, 16, 17, 18]. But the tracing function is not complete if the system does not provide unconditional tracing [2, 8, 19, 20, 21, 22, 23, 24, 25, 26, 27].

In our opinion, for crime prevention, the unconditional tracing is necessary, however, the existence of TTP unavoidably threatens the anonymity of honest users. So it is reasonable that the system can be independent of TTP, but at the same time, the scheme can easily be slightly modified to select the available tracings depending on the actual circumstances.

In addition, if the user loses E-cash, he should find some way to get his money back if the lost E-cash cannot be spent by anyone. And it could be achieved if bank provides lost-coin tracing (in the validity period of coins), which is ignored by almost all schemes.

There are 2^l coins in the customer's wallet in compact E-cash system. So in such system, double-spending tracing and unconditional tracing should include coin-tracing. **Table 1** shows the property of the practical and complete tracing.

Table 1. The property of the practical and complete tracing

Tacing types	Double-spending tracing		Unconditional tracing		Loss tracing
	tracing spender	tracing coins	tracing owner	tracing coins	tracing coins
Provider	Bank		TTP		Bank
Condition	Double-spending		None (only in crime cases)		Valid loss-register for unspent coins
Module	Necessary		Optional		Optional

In fact, the "practical" and "complete" tracing is not easy for existing schemes. Moreover, the recoverable E-cash scheme is presented in [20, 28, 29] for sloving the coin-lost problem, but there were some unsolved problems. In our [30], we clarify all there problems and provide a solution. But to achieve the practical and complete tracing in compact E-cash, there is a new problem.

Part II: the Efficiency problem in compact E-cash

The significant contribution of compact E-cash [8] is that the computation complexity of withdrawing 2^l coins is $O(1)$, and the storage space of the 2^l e-coins is just $O(l)$ bits, while before it is proposed, the E-cash schemes have to withdraw 2^l coins or store them with $O(2^l)$ complexity. Then some other interesting compact E-cash schemes [21, 23, 31] are designed based on different cryptographic techniques. But good efficiency, which is the very important for compact E-cash, does not attract enough attention in the designing. After all, achieving efficient system is the main goal of compact E-cash.

The main efficiency problem is that tracing coins without TTP results in poor efficiency [8, 21, 23, 31]. When the user double-spends E-cash, tracing him is necessary. However, tracing his coins is very important, since the doulbe-spender has at least 2^l coins so that he can cheat more times if system cannot trace his remaining coins. As we discussed, double-spending tracing is the bank's task, not the TTP's. Obviously, if TTP performs every tracing, it seriously threatens the anonymity of honest users. However, when double-spending, it is not easy for existing schemes that bank traces double-spender's coins without TTP.

[21, 23] and the system 1 of [8] only provide the double-spender tracing. The system 2 of [8] and [31] provide coin-tracing. However, when adding this coin-tracing without TTP, their systems [8, 31] become inefficient. For example, when the system 2 of [8] adds coin-tracing, the inefficiency is pronounced —— in the system 1, the withdrawal uses 12 multi-based exponentiations, while in the system 2, it uses 810 multi-based exponentiations and 300

bilinear pairings. At the same time, in the system1, spending one coin uses 40 multi-based exponentiations, while in the system 2, it uses 778 multi-based exponentiations.

In existing compact E-cash schemes [8, 21, 23, 31], there are some difficulties in designing coin-tracing without TTP for doulbe-spending. And we will clarify it. All compact E-cash schemes are constructed based on zero-knowledge proof. In short, when spending a coin from the compact E-cash wallet, the customer shows the shop a pseudorandom function with the wallet parameters and proves the pseudorandom function is constructed correctly using zero-knowledge proof. So when the customer doulbe-spends a coin, the bank needs the wallet parameters as the input of the pseudorandom function so as to trace the remaining coins from the double-spender. But without a TTP, for recovering the secret wallet parameters, [8] uses the verifiable encryption and [31] uses the accumulator. Unfortunately, these cryptographic techniques result in poor efficiency.

Tacing without TTP implies that the information of secret parameters is leaked when double-spending. But for the anonymity of honest customers, the knowledge of secret parameters is proven using zero-knowledge proof. Therefore, to solve the problem of recovering secret parameter, our idea is reconstructing the zero-knowledge proof [32]. And the new zero-knowledge proof can leak the information of the proven parameters when it is used twice to prove the same knowledge (double-spending), but it has the perfect zero-knowledge property when it is used once (normal-spending). That is to say, it guarantees the anonymity in normal case and revokes the anonymity in abnormal case.

To the best of our knowledge, when the customer double-spends a coin, the efficiency problem caused by tacing coins without TTP had not been solved in existing compact E-cash schemes. And we solve it —— achieving coin-tracing without TTP, our computation complexity of withdrawing E-cash is $\boldsymbol{O(1)}$, while it is $\boldsymbol{O(k)}$ in [8] and it is $\boldsymbol{O(2^{2n})}$ in [31].

Our main contributions:

- ✧ We present a compact E-cash scheme with practical and complete tracing.
- ✧ We provide a solution to the efficiency problem in compact E-cash system when tracing double-spender's coins without TTP.
- ✧ The new knowledge proof is zero-knowledge to verifier in normal-spending, while it leaks the proven knowledge in double-spending.

In the following parts, Section 3 provides preliminaries. Section 4 presents the system model. In section 5, we describe the tracing mode. Section 6 provides the details of the proposed scheme. The key security proofs are presented in section 7. And in section 8, we compare the system efficiency of schemes. Finally, the conclusion is provided in section 9.

3. Preliminaries

3.1 Assumptions

Assumption 1 (S-RSA Assumption) [33]**.** *Let $n=pq$ be an RSA-like modulus and $z\in Z^*_n$. It is hard to compute $u\in Z^*_n$ and integer $e>1$ such that $z \equiv u^e \pmod n$.*

Assumption 2 (DDH Assumption) [34]**.** *Let $G=\langle g\rangle$ be a cyclic group generated by g of order $u=\#G$ with $\lceil log_2(u)\rceil=l_G$. Given $(g, g^x, g^y, g^z) \in G^4$, it is hard to decide whether g^z and g^{xy} are equal.*

Assumption 3 (q-DDHI Assumption) [8]**.** *Let $G=\langle g\rangle$ be a cyclic group generated by g. Given the elements $(g, g^x,\ldots, g^{(x^q)}) \in (G^*)^{q+1}$, it is still hard to decide whether $g^{1/x}$ and a random element in G are equal.*

The ***(t, q, ε)*-DDHI** assumption means that there is no ***t***-time algorithm which has the

advantage at least ε to break the ***q*-DDHI** assumption.

3.2 Zero-knowledge Proof [35, 36]

Definition 1. *Let $A=\{A(x)\}_{x\in L}$ and $B=\{B(x)\}_{x\in L}$ be two ensembles of variables indexed by strings $x\in L$, where $L\in\{0,1\}^*$. A and B are statistically indistinguishable, if for any polynomial $p(.)$ and any $x\in L$ it holds that $\sum_{t\in\{0,1\}^*}|Prob(A(x)=t)-Prob(B(x)=t)|<1/p(|x|)$.*

Definition 2. *Protocol (P,V) is statistical zero-knowledge, if $\{[P, V](x)\}_{x\in L}$ and $\{S_{[P,V]}(x)\}_{x\in L}$ are statistically indistinguishable, where V can be any probabilistic polynomial-time verifier and $S_{[P,V]}$ is a probabilistic polynomial-time simulator which can simulate the protocol (P,V).*

Definition 3. *$(c,s)\in\{0,1\}^k\times\pm\{0,1\}^{\varepsilon(l_G+k)+1}$ satisfying $c=H(y\|g\|g^s y^c\|m)$ is a signature of knowledge (SPK) on the message m, which uses the knowledge of discrete logarithm of y to the base g. And $\boldsymbol{SPK}(\alpha: y=g^{\alpha})(m)$ denotes it.*

Using the knowledge of discrete logarithm, i.e., $x=\log_g y$, $\boldsymbol{SPK}(\alpha: y=g^{\alpha})(m)$ is computed as follows. After choosing $r\in\pm\{0,1\}^{\varepsilon(l_G+k)}$, the signer computes $c=H(y\|g\|g^r\|m)$ (i.e., **challenge**) and $s=r-cx$ (i.e., **challenge-response**), and g^r is the **commitment** to prove that the signer knows $x=\log_g y$. Here $H(\ ):\{0,1\}^*\rightarrow\{0,1\}^k$ denotes a hash function. The interactive protocol of ***SPK*** performed by prover and verifier is the zero-knowledge proof of the knowledge of $x=\log_g y$ provided by prover, which is denoted by ***PK***( ) [8].

3.3 Pseudorandom Function [37]

Generating Key: Choose the secret key $SK\in_R Z_p^*$, and the public key $PK=g^{SK}$.

Verifiable Random Function: $F_{SK}(x)=e(g,g)^{1/(x+SK)}$ is a verifiable random function and $p_{SK}(x)=g^{1/(x+SK)}$ is the proof of correctness of it.

Verification: Verify $e(g^x\cdot PK, p_{SK}(x))=e(g,g)$ and $F_{SK}(x)=e(g,p_{SK}(x))$. If it is true, $F_{SK}(x)$ is proven to be generated correctly.

So $F_{SK}(x)=h^{1/(x+SK)}$ is viewed as a pseudorandom function (**PRF**) [37]. $F_{SK}(x)$ is an $(s'(k), \varepsilon'(k))$ secure **PRF** if no one can break the pseudo randomness property with $\varepsilon'(k)$ advantage in $s'(k)$ time.

4. Security Model of the E-cash System

4.1 Syntax

There are four kinds of entities: **C** (customer), **B** (bank), **S** (shop) and **T** (TTP or trusted authorities). There are the polynomial time algorithms or protocols: B/T/C Setup, Withdraw, Pay, Deposit, UnconditionallyTrace, LossCoinTrace, DoubleSpendTrace. P(E1(x_1), E2(x_2)) denotes a protocol between E1 and E2, and E1's input is x_1 and E2's is x_2.

- ✧ B/T/C Setup(params). The algorithm outputs B/T/C's private/public key pair $(PK_B,SK_B)/(PK_T,SK_T)/(PK_C,SK_C)$, here C includes S.
- ✧ Withdraw(C(SK_C, PK_B), B(PK_C, SK_B)). It allows C to withdraw a certain amount of E-cash. C receives X (E-cash), i.e. an identifier I and a proof of validity Π, or one error message $\perp$. B receives the view of the protocol (we call this V_{PKC}) or one error message $\perp$ if V_{PKC} is not proven from C with PK_C.
- ✧ Spend(C(X, PK_B, SK_C), S(PK_S)). It allows C to pay e-coin from X to S. S receives the proof π of payment with *aux* (auxiliary information) or one error message $\perp$ if π is invalid. C receives the updated wallet X' if the payment is accepted by S.

- ✧ Deposit(S(PK_S, π, aux), B(PK_B)). It allows S to send (PK_S, π, aux) to B for deposit. After verifying (PK_S, π, aux), B adds (π, aux) to spent records. But B outputs one error message $\perp$ if (PK_S, π, aux) is invalid or B executes the DoubleSpendTrace algorithm if the coin is double-spent.
- ✧ UnconditionallyTrace(B(V_{PKC}, π, aux), T(SK_T)). It is an algorithm which allows T to trace any e-coin from X and any owner when the transaction is involved in crimes. T outputs particular e-coin tracing information $I_{e\text{-}coin}$, owner tracing information I_{owner} and proof P_1 which proves the connection between the e-coin from X and C, but if the input is illegal, T outputs an error message $\perp$.
- ✧ LossCoinTrace(C(SK_C), B(PK_C, V_{PKC})). It allows C to register his lost coin/wallet in system. B outputs loss tracing information I_{loss} and a proof P_2 which proves the validity of the loss register without T, but if the proof of owning the lost-coin/wallet can't be provided, B outputs an error message $\perp$; If the lost coin was spent, B outputs the spent proof P_{spent}.
- ✧ DoubleSpendTrace(π_1, π_2). With double-spending proofs related to one coin, B executes the algorithm and outputs PK_C of the double-spender, the tracing information I_X of the coins from X and the proof P_3 which proves that C with PK_C is the double-spender and the traced coins with I_X is owned by the double-spender, but if π_1=π_2, the algorithm outputs an error message $\perp$.

4.2 Security Definitions

— ***Balance***. In the Withdraw protocol, Withdraw_m is the middle of the protocol. m_1 is the first message sent by C and b_1 is B's state information when m_1 is received. The balance property indicates that:

- There are the efficiently decidable language l_S and the extractor $\mathcal{E}^{X\text{-}Y(A)}_{\text{Withdrawm},ls}$(*params*, *auxext*, PK_C, m_1, b_1) [8] such that for all b_1 and m_1, which extracts $w = (\Theta_1, \ldots, \Theta_n, sec_w)$ such that $(m_1, b_1, w) \in l_S$ whenever the probability that B accepts in the Withdraw_m part of the protocol is non-negligible, where Θ is the serial number of E-cash (e-coin). In the case, the extractor outputs $(m_1, b_1, w) \in l_S$.
- With (*params*, PK_B), the adversary $\mathcal{A}$ plays the game as follows: $\mathcal{A}$ performs Withdraw/Deposit protocols with B polynomial times. ($\mathcal{A}$ simulates running Spend protocol with itself.) $(m_{1,i}, b_{1,i}, w_i) \in l_S$ is the output of $\mathcal{E}^{X\text{-}Y(A)}_{\text{Withdrawm}}$ (*params*, *auxext*, PK_{Ci}, $m_{1,i}$, $b_{1,i}$) if the ith Withdraw protocol is successful, where $w_i = (\Theta_{i,1}, \ldots, \Theta_{i,n}, sec_{wi})$ are 2^l serial numbers belonging to PK_{Ci}. $A_f = \{\Theta_{i,j} \mid 1 \le i \le f, 1 \le j \le n\}$ is the set of all serial numbers after performing Withdraw protocol f times. $\mathcal{A}$ wins this game if for some f, B accepts a coin with one $\Theta \notin A_f$ in Deposit protocol. The secure E-cash scheme requires that no probabilistic polynomial-time (PPT) $\mathcal{A}$ can win the game with non-negligible probability.

— ***Complete-Tracing***. It guarantees that no PPT $\mathcal{A}$ has a non-negligible probability of winning the following game:

On input (*params*, PK_B), $\mathcal{A}$ performs Withdraw/Deposit protocols with B polynomial times. ($\mathcal{A}$ also simulates running Spend protocol with itself.) A_i be the set of serial numbers which belong to C_i with PK_{Ci}. $\mathcal{A}$ wins the game if any of the following cases occurs.

- On input (B(V_{PKCi}, π, aux), T(SK_T)), the algorithm UnconditionallyTrace can not output e-coin tracing information $I_{e\text{-}coin\text{-}i}$, owner tracing information $I_{owner\text{-}i}$ or proof P_{1i} which proves the coins from X_i owned by C_i;

- On input ($C_i(SK_{Ci})$, $B(PK_{Ci}, V_{PKCi})$), the protocol LossCoinTrace can not output loss tracing information $I_{loss\text{-}i}$ or a proof P_{2i} which proves the validity of the loss register;
- On input (π_1, π_2) of the same coin, in some Deposit protocol, B accepts the same coin spent twice with serial number $\Theta_{i,j} \in A_i$ twice, i.e., B accepts ($\Theta_{i,j}$, π_1, j) and ($\Theta_{i,j}$, π_2, j) and cannot find the double-spending behavior;
- On input (π_1, π_2) of the same coin, DoubleSpendTrace cannot output double-spender's public key or the related information of the coins in X_i from double-spender for tracing;
- On input (π_1, π_2) of the same coin, DoubleSpendTrace cannot output the proof P_{3i}, which proves that C_i with PK_{Ci} is double-spender or all the traced coins are owned by the double-spender C_i.

— *Anonymity of customer*. $\mathcal{A}$ generates B's pulic key PK_B and then plays the following games arbitrarily:

Game R_1

- (1) *PK* of C_i In the phase, $\mathcal{A}$ can request and receive any PK_{Ci} (public key) of C_i, that is generated in Setup phase.
- (2) Withdrawing with C_i $\mathcal{A}$ performs Withdraw protocol with C_i: Withdraw(C(SK_{Ci}, PK_B, n), $\mathcal{A}$(*state*, n)); X_j is C_i's output after the j'th withdrawing, and X_j could be the error message, e.g., X_j is invalid.
- (3) Spending from X_j $\mathcal{A}$ performs Spend protocol with C_i if X_j is valid: Spend(C(X_j, PK_B, SK_{Ci}), $\mathcal{A}$(*state*)). And $\mathcal{A}$ cannot request C_i to spend the same coin more than once.

Game R_2 The phases (1) and (2) are the same as them in Game R_1, but in the phase (3), $\mathcal{A}$ performs Spend protocol with a simulator $S^{X\text{-}Y(A)}$ (*params*, *auxsim*, PK_B).

Without the knowledge of x about Game R_x, $\mathcal{A}$ guesses x' of Game $R_{x'}$, where $x, x' \in \{0,1\}$. Anonymity of customer means that $\delta = (Prob(x' = x) - 1/2)$ is negligible for the PPT $\mathcal{A}$.

— *Strong Exculpability*. $\mathcal{A}$ has the knowledge of secret key SK_B and SK_C of collusive C and can act as B or S or collusive C in sytem protocols. $\mathcal{A}$ can choose any honest C_i with public key PK_{Ci} and perform system protocols with him arbitrarily. If any of the following cases occurs, $\mathcal{A}$ wins this game.

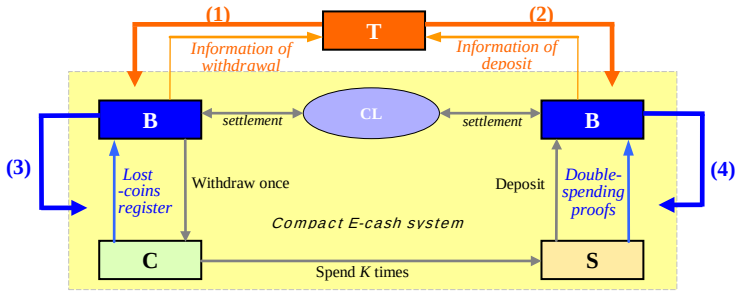
- $Case_1$: If C_i is honest, i.e., C_i never double-spends one coin, DoubleSpendTrace algorithm outputs (P_{3i}, PK_{Ci}, I_{Xi});
- $Case_2$: If C_i is honest, $\mathcal{A}$ gets some valid spending proof (π_i, aux_i, Θ_i) where Θ_i is the serial number of some coin, but C_i has not spent the related coin;
- $Case_3$: If C_i spends one coin twice, $\mathcal{A}$ gets (π_{1i}, aux_{1i}, Θ_{di}) and (π_{2i}, aux_{2i}, Θ_{di}) respectively. Then (P_{3i}, PK_{Ci}, I_{Xi}) is the output of DoubleSpendTrace algorithm on input (π_{1i}, π_{2i}). Then DoubleSpendTrace algorithm outputs (P_{3i}', PK_{Ci}, $I_{Xi'}$, Θ'_{di}), but C_i does not spend the coin with Θ'_{di} twice;
- $Case_4$: If C_i spends one coin twice, $\mathcal{A}$ gets (π_{1i}, aux_{1i}, Θ_{di}) and (π_{2i}, aux_{2i}, Θ_{di}) respectively. Then (P_{3i}, PK_{Ci}, I_{Xi}) is the output of DoubleSpendTrace algorithm on input (π_{1i}, π_{2i}). Then $\mathcal{A}$ gets some valid spending proof (π'_i, aux'_i, Θ'_i), but C_i has not spent the related coin.

Strong Exculpability means the probability that the PPT $\mathcal{A}$ wins the above game is negligible.

5. Tracing Mode of the Proposed Compact E-cash Scheme

In the proposed scheme, only when crimes occur, TTP has the power to execute unconditional

coin-tracing (according to the information from withdrawal protocol provided by bank) and unconditional owner-tracing (according to the information from deposit provided by shop). When customer registers for his lost coins, bank executes the lost-coin tracing without TTP. When customer spends a coin twice, bank executes the double-spender/coin tracing without TTP. From **Fig. 2**, we can see that the system with the basic tracing functions can be independent of TTP, that is to say, unconditional tracing is optional in our system.



T : TTP B : bank C : customer S : shop CL : clearing house
(1): Unconditional coin-tracing (3): Lost-coin tracing
(2): Unconditional owner-tracing (4): Double-spender/coin-tracing

Fig. 2. Compact E-cash system providing complete and practical tracing

6. Compact E-cash with Practical and Complete Tracing

6.1 Overview of the Proposed Scheme

✧ **Withdrawal Protocol**

C and B generate the wallet parameters (e_1, e_2, x) which are used to generate 2^l coins, and B signs them using CL signature [38]. Then to achieve loss tracing and unconditional tracing, C provides two ElGamal encryptions—— $\text{ElGamal}_{\text{PK}_C}(g^{e_2})$ encrypts g^{e_2} using C's public key PK_C and $\text{ElGamal}_{\text{PK}_T}(g^{e_2})$ encrypts g^{e_2} using T's public key PK_T. Also, C provides the knowledge proof (*SPK*) to prove the encryptions are generated correctly.

✧ **Payment Protocol**

C performs the protocol with S to spend one of coins from C's wallet.

(1) To prove the validity of coin, C provides the zero-knowledge proof of (e_1,e_2,x) to prove that the coin spent in this protocol is from some signed wallet.

(2) To achieve loss tracing and unconditional tracing, C computes $T_2 = g^{H(J\|r)e_2} \pmod{n_T}$ with the random input r, where n_T is RSA modulus and T has its factor knowledge, integer $J \in [0, 2^l\text{-}1]$. C also provides the zero-knowledge proof to prove that T_2 is generated correctly.

(3) To prevent double-spending, C provides the serial number of coin, i.e., $\Theta = \text{PRF}(e_2, J)$, where PRF( ) is a pseudorandom function with the secret seed e_2 and the public input J. The Θ records the spent coins for B and the integer $J \in [0, 2^l\text{-}1]$ records the spent coins for C. So C's wallet contains 2^l coins. Also, C provides the zero-knowledge proof to prove that Θ is generated correctly.

(4) To efficiently achieve double-spending tracing without TTP, C constructs a special knowledge proof of e_1, i.e., $PK_\Theta(e_1)$, which is related to Θ. The special property of $PK_\Theta(e_1)$ is that if showing $PK_\Theta(e_1)$ with the same Θ twice, i.e., spending the same coin twice, the knowledge of (e_1,e_2) is leaked. The details of $PK_\Theta(e_1)$ are provided in the following part.

✧**Deposit Protocol**

S sends B the information generated by C in the above payment protocol. B verifies it as S does in payment protocol and makes sure that the coin is spent only once.

✧ **Loss Register Protocol**

For registering to trace his lost coins, C sends B the information of remaining coins, i.e., LR_x, which is just shown in loss register protocol so that it does not affect C's anonymity when spending the remaining coins. Then B sends C the $\text{ElGamal}_{PK_C}(g^{e_2})$ and the related zero-knowledge proof of it. After verifying the related zero-knowledge proof, C believes that the $\text{ElGamal}_{PK_C}(g^{e_2})$ was generated by himself in withdrawal protocol and then uses his private key to decrypt $\text{ElGamal}_{PK_C}(g^{e_2})$ so as to get the g^{e_2}.

✧ **Practical Complete Tracing**

· Unconditional tracing

Unconditional coin-tracing : Getting the information of withdrawal from bank, T uses the private key to decrypt $\text{ElGamal}_{PK_T}(g^{e_2})$ and publishes g^{e_2}. In payment protocol, $T_2 = g^{H(J\|r)e_2} \pmod{n_T}$ will be provided to S with the shown J and r, so T_2 can be identified if g^{e_2} is known, that is to say, all the coins can be traced.

Unconditional owner-tracing: Getting the information of deposit from shop, T uses the factor knowledge of n_T to compute the inverse of $H(J\|r)$ so as to get g^{e_2} from T_2, then T can identify the owner according to withdrawal database.

· Double-spending tracing

Double-spender-tracing: If C double-spends a coin, he has to use the same J more than once, that is, he has to show the same $\Theta = \text{PRF}(e_2, J)$ more than once, so it can be found out by B. As mentioned earlier, if showing the special knowledge proof $PK_\Theta(e_1)$ with the same Θ twice, (e_1,e_2) will be leaked. Since e_1 is used as tracing information of customer in withdrawal protocol, the double-spender can be traced.

Double-spender's coin-tracing: With the leaked e_2, every $\Theta=\text{PRF}(e_2, J)$ is computed and published for $J\in[0, 2^l\text{-}1]$ so that the coins from double-spender cannot be spent anymore.

· Lost-coin tracing

After loss register, B publishes g^{e_2}. In payment protocol, $T_2 = g^{H(J\|r)e_2} \pmod{n_T}$ is provided to S with the shown J and r, S can identify T_2 after g^{e_2} is published so that the lost coins cannot be spent by others.

Then, we provide the detailed scheme.

6.2 System Setup

Let $\varepsilon>1$, k, l_s, l_p and l_{ps} be security parameters. λ_1, λ_2, γ_1 and γ_2 denote bit-length satisfying $\lambda_1>\varepsilon(\lambda_2+k)+2$, $\gamma_1=l_{ps}+2$, $\gamma_1>\varepsilon(\gamma_2+k)+2$, $\gamma_2,\lambda_2>k$. And $\Lambda = [2^{\lambda_1}\text{-}2^{\lambda_2}, 2^{\lambda_1}+2^{\lambda_2}]$ and $\Gamma = [2^{\gamma_1}\text{-}2^{\gamma_2}, 2^{\gamma_1}+2^{\gamma_2}]$. $H(\ )$ is a hash function: $\{0, 1\}^* \rightarrow \{0, 1\}^k$.

B (bank)'s Setup:

Select random secret l_p-bits primes p' and q', and $p = 2p'+1$, $q = 2q'+1$ are primes. Provide the zero-knowledge proof to prove that $n = pq$ is the product of two safe primes [39].

T (trusted authority)'s Setup:

Select a random l_{ps}-bits prime n_s, and $l_{ps}=\varepsilon(\lambda_2+k)+2k+2>\lambda_1$. Select two secret l_p-bits primes p'' and q'' such that $p_T = 2p''+1$ and $q_T = 2q''+1$ are primes. Prove that $n_T = p_Tq_T$ is the product of two safe primes [39] with the zero-knowledge proof. Choose the elements a, a_0, a_1, a_2 of $QR(n)$ of order $p'q'$, g of $QR(n_T)$ of order $p''q''$ [35]. $K_1 \in_R I_{l_s}$ and set $h=g^{K_1} \pmod{n_T}$.

C (customer)'s Setup:

$SK_C \in_R I_{l_s}$ and set $PK_C = g^{SK_C} \pmod{n_T}$.

The system public parameters are $PK=(n,n_s,n_T,a,a_0,a_1,a_2,g,h)$, B's private key is p', T's private key is (p'',K_1), and C's private key is SK_C.

6.3 Withdrawal Protocol

As **Fig. 3** presents, C and B generate cash parameters (x, e_1, e_2) together, then B computes the CL signature [38] (i.e., (A, e)). As mentioned in section 6.1, (A_1, A_2) is $\text{ElGamal}_{PK_C}(g^{e_2})$ which encrypts g^{e_2} using C's public key PK_C for loss tracing and (A_2, A_3) is $\text{ElGamal}_{PK_T}(g^{e_2})$ which encrypts g^{e_2} using T's public key h for unconditional tracing. Then C gets (x, e_1, e_2, A, e, r_C) satisfying $A^e \equiv a_0 a^x a_1^{e_1} a_2^{e_2} \pmod{n}$, and r_C is the series number of wallet. W and W_1 guarantee the correctness of computations, and W uses the method in [35] to prove the value range of e_1.

C	B
$x,e_1 \in_R [2^{\lambda_1}-2^{\lambda_2}, 2^{\lambda_1}+2^{\lambda_2}]$, $e_{2C} \in_R [0, n_s-1]$	
$T_C = a^x \cdot a_1^{e_1} \cdot a_2^{e_{2C}} \pmod{n}$, $r_C \in_R I_{l_s}$	
$W=SPK(\alpha,\beta,\gamma,\chi: PK_C=g^\chi \wedge T_C=a^\alpha \cdot a_1^\beta \cdot a_2^\gamma \wedge \beta \in [2^{\lambda_1}-2^{\varepsilon(\lambda_2+k)}, 2^{\lambda_1}+2^{\varepsilon(\lambda_2+k)}])(r_C)$	
$\xrightarrow{T_C,\ r_C,\ W,\ PK_C}$	
	Verify W, choose a prime $e \in_R \Gamma$, $e_{2B} \in_R [0, n_s-1]$ Compute $A \equiv (a_0 \cdot T_C \cdot a_2^{e_{2B}})^{1/e}$
$\xleftarrow{A,\ e,\ e_{2B}}$	
Verify $e_{2B} \in_R [0, n_s-1]$, $e \in \Gamma$	
Verify $A^e \equiv a_0 T_C \cdot a_2^{e_{2B}} \equiv a_0 \cdot a^x \cdot a_1^{e_1} \cdot a_2^{e_{2C}+e_{2B}}$ $\equiv a_0 \cdot a^x a_1^{e_1} a_2^{e_2} \pmod{n}$	
$A_1=g^{e_2} PK_C^{e_1} \pmod{n_T}$ $A_2=g^{e_1} \pmod{n_T}$ $A_3=g^{e_2} h^{e_1} \pmod{n_T}$	
$W_1= SPK(\alpha,\beta,\gamma: A^e = a_0 a^\alpha a_1^\beta a_2^\gamma \wedge A_1=g^\gamma PK_C^\beta \wedge A_2=g^\beta \wedge A_3=g^\gamma h^\beta)$	
$\xrightarrow{A_1,\ A_2,\ A_3,\ W_1}$	
	Verify W_1, Store r_C, W, A_1, A_2, A_3, W_1 Record a debit of 2^l coins for C's account

Fig. 3. Withdrawal protocol ($\in_R$ denotes choosing at random)

6.4 Payment Protocol

As we sketched in the section 6.1,

(1) To prove the validity of coin anonymously, C conceals the wallet $T_1=Ah^w \pmod{n}$ [8] and provides the zero-knowledge proof of it—— since $A^e \equiv a_0 a^x a_1^{e_1} a_2^{e_2} \pmod{n}$, C provides $PK_1(e, x, e_1, e_2, ew: a_0 = T_1^e / (a^x a_1^{e_1} a_2^{e_2} h^{ew}))$ as the zero-knowledge proof of the wallet ownership. For simplifying denotation, we directly use the parameter name in $PK(\ )$.

(2) To achieve loss tracing and unconditional tracing, C computes $T_2 = g^{H(J \| r) e_2} \pmod{n_T}$ with the random r. C also provides the zero-knowledge proof $PK_2(e_2: T_2 = g^{H(J \| r) e_2})$.

(3) To prevent double-spending, C provides the serial number of coin $\Theta = a_{1J}^{(J+e_2)^{-1} \pmod{n_s}} \pmod{n}$ with $J \in [0, 2^l-1]$ and $PK_3(e_2: \Theta = a_{1J}^{(J+e_2)^{-1} \pmod{n_s}})$ proving the correctness of Θ.

Since $\Theta^{(J+e_2)} = a_{1J}^{(J+e_2)^{-1} \pmod{n_s} (J+e_2)} = a_{1J}^{t \cdot n_s+1} \pmod{n}$ $t \in Z$

$\Theta^{e_2} = a_{1J}^{t \cdot n_s+1} / \Theta^J \pmod{n}$

That is to say, $PK_3(e_2: \Theta = a_{1J}^{(J+e_2)^{-1} \pmod{n_s}}) = PK(t, e_2: T = a_{1J}^{t \cdot n_s+1} \wedge \Theta^{e_2} = T/\Theta^J)$.

(4) To achieve double-spending-tracing without TTP, C constructs a special knowledge proof $PK_\Theta(e_1)$. It leaks (e_1,e_2) if showing $PK_\Theta(e_1)$ with the same Θ twice. Now we clarify it.

According to Definition 3, to prove the discrete logarithm knowledge in $y=g^{\alpha}$, first, prover shows verifier the **commitment** g^{r} with a random integer r, after receiving the random tr from verifier, gets the **challenge** $c=H(y\|g\|g^{r}\|tr)$ and then computes the **challenge-response** $s=r-cx$. Therefore, $g^{r}=g^{s}y^{c}$ (or $c=H(y\|g\|g^{s}y^{c}\|tr)$) is the knowledge proof of $y=g^{\alpha}$.

The idea of constructing $PK_{\Theta}(e_1)$ is to replace r with $(J+e_2)^{-1} \pmod{n_s}$ in the knowledge proof of e_1. If showing $PK_{\Theta}(e_1)$ with the same Θ twice, there are two **challenge-response** equations: $s_{\sigma}=(J+e_2)^{-1} \pmod{n_s}-c(e_1-2^{\lambda_1})$ and $s'_{\sigma}=(J+e_2)^{-1} \pmod{n_s}-c'(e_1-2^{\lambda_1})$, so it is easy to compute (e_1,e_2) from the two **challenge-response** equations. For achieving it, C computes $d_1=T_1^{r_1}/[a^{r_2}a_1^{(J+e_2)^{-1}(\mathrm{mod}\ n_s)}a_2^{r_3}h^{r_4}]$ as the **commitment** of $PK_1(e,x,e_1,e_2,ew: a_0=T_1^{e}/(a^{x}a_1^{e_1}a_2^{e_2}h^{ew}))$, that is to say, $(J+e_2)^{-1} \pmod{n_s}$ replaces r in d_1. And C must provide the zero-knowledge proof $PK_4(\delta: \Theta=a_{1J}^{\delta} \wedge d_1=T_1^{r_1}/(a^{r_2}a_1^{\delta}a_2^{r_3}h^{r_4}))$. So PK_3 and PK_4 guarantee that one **challenge-response** of PK_1 is $s_{\sigma}=(J+e_2)^{-1} \pmod{n_s}-c(e_1-2^{\lambda_1})$.

The special design changes the construction of zero-knowledge proof. So we must prove:

- the security of construction of zero-knowledge proof is preserved;
- $(J+e_2)^{-1} \pmod{n_s}$ can be used as a pseudorandom function in this construction;
- the special knowledge proof has the zero-knowledge property.

The proofs of the above security properties are provided in the section 7.

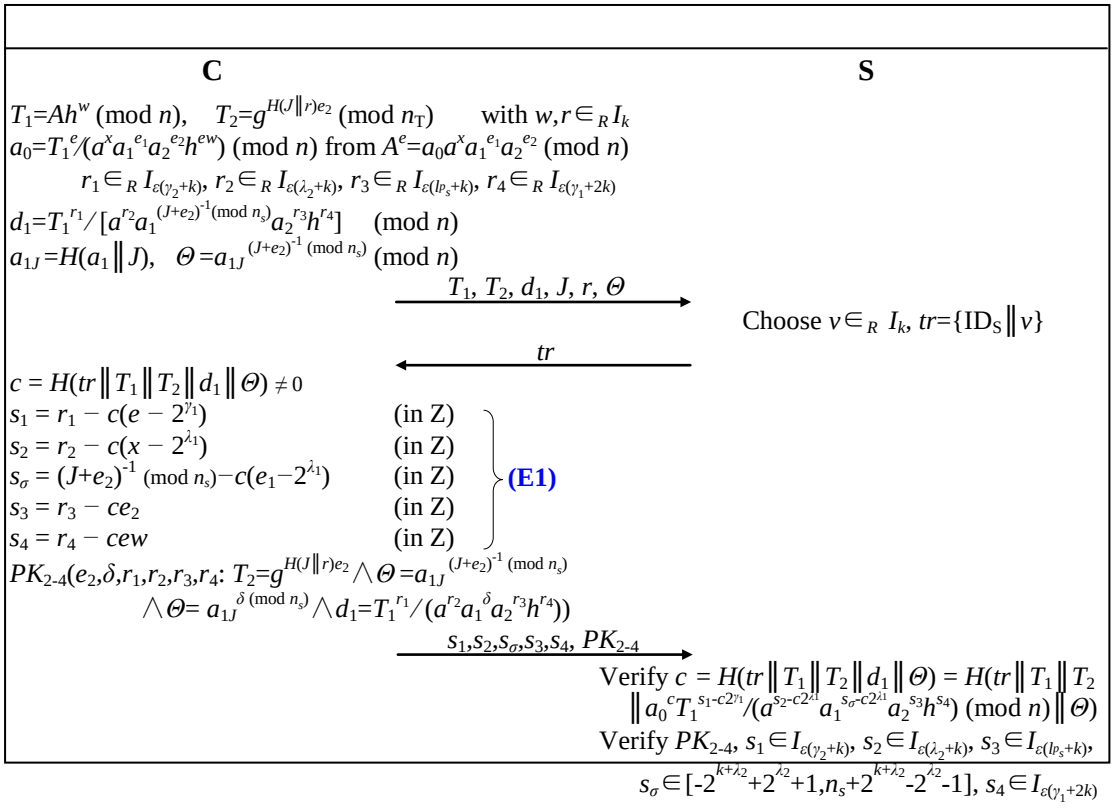


Fig. 4. Payment protocol (I_d denotes $\{0,1\}^{d}$)

For presenting our idea clearly, only the **commitment** and **challenge-response** of $PK_1(\ )$ are provided, since $PK_{\Theta}(e_1)$ is embedded in it. And $PK_2(\ )$~$PK_4(\ )$ are common zero-knowledge proofs, so the **commitment** and **challenge-response** of them are omitted in **Fig. 4** (we just use $PK_{2\text{-}4}$ to denote the process). In fact, all the **commitments** are provided in the commitment stage and all the **challenge-responses** are computed in the challenge-response stage.

6.5 Loss Register Protocol

When C loses the E-wallet with the series number r_{Cx}, he sends $(r_{C1},\ldots,r_{Cx-1}, r_{Cx+1},\ldots,r_{Cn})$ of his remaining E-wallet to B, and it is shown in **Fig. 5**. As r_{Ci} is not shown when C spends E-cash, the anonymity of C and the unlinkablity of spending will not be influenced. Note that only after B confirms the quantity of all unspent coins in system, that is to say, after a period, C gets the refund from B. In addition, B cannot refuse to perform loss-tracing unless B can provide the payment proof generated before B publishes loss-tracing information in system.

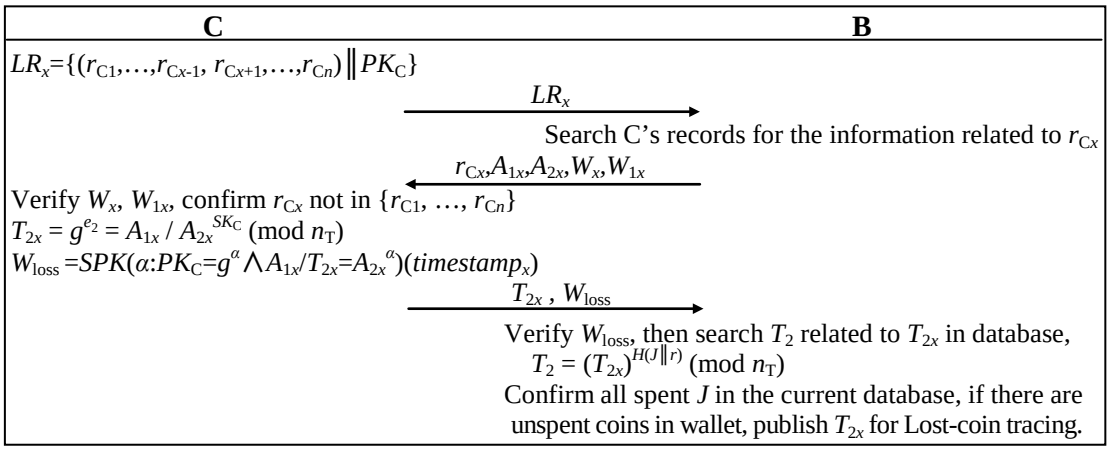


Fig. 5. Loss Register Protocol

6.6 Deposit Protocol

S sends B the payment proof (i.e., $Proof_{payment} = (T_1,T_2,J,r,\Theta,tr,c,s_1,s_2,s_\sigma,s_3,s_4,PK_{2\text{-}4})$). B verifies $Proof_{payment}$ as S does in **Fig. 4**, then B searches Θ in database. If there is the same Θ with the same J and a different c, it indicates that the $Proof_{payment}$ is generated by some double-spender. Therefore, it results in Double-spending tracing. However, if there is the same (Θ, J, c) in B's database, it indicates that S deposits this coin twice, and B will abort it. Otherwise, B records the $Proof_{payment}$ and terminates this protocol.

6.7 Complete Tracing

Double-spending tracing

If C spends the same coin twice, B can trace the double-spender and his coins.

✧ Double-spender tracing from Bank

$$\begin{cases} s_\sigma = (J+e_2)^{-1} \pmod{n_s} - c(e_1 - 2^{\lambda_1}) \\ s'_\sigma = (J+e_2)^{-1} \pmod{n_s} - c'(e_1 - 2^{\lambda_1}) \end{cases}$$

B can compute e_1 and e_2 from the equation set.

$$\begin{cases} e_1 = (s_\sigma - s'_\sigma)/(c'-c) + 2^{\lambda_1} \\ e_2 = [s_\sigma + c(s_\sigma - s'_\sigma)/(c'-c)]^{-1} \pmod{n_s} - J \end{cases}$$

Since $A_2 = g^{e_1} \pmod{n_T}$, $A_2 = g^{(s_\sigma - s'_\sigma)/(c'-c) + 2^{\lambda_1}} \pmod{n_T}$……**(e1)**

Therefore, the double-spender with PK_C will be found according to A_2 stored in withdrawal database. $W_{double} = (\Theta, J, tr, tr')$ proves that some C double-spends, and $(W_{double}$,**(e1)**,$W,W_1)$ prove that C with PK_C double-spends.

✧ Double-spender's coin tracing from Bank

Since $\Theta = a_{1J}^{(J+e_2)^{-1} \pmod{n_s}} \pmod{n}$, $\Theta^* = a_{1J}^{\{J^* + [s_\sigma + c(s_\sigma - s'_\sigma)/(c'-c)]^{-1} \pmod{n_s} - J\}^{-1} \pmod{n_s}} \pmod{n}$ …**(e2)**

So B can compute Θ^* for every J^* and identify all e-coins in the double-spender's e-wallet.

And (W_{double}, **(e2)**) prove that the coins belong to a double-spender.

Unconditional tracing

When E-cash is connected with crimes, T can do the following tracing.

✧ Unconditional owner-tracing from T

T gets J, r and T_2 from payment, and $T_2 = g^{H(J\|r)e_2} \pmod{n_T}$. Then T uses the private key p'' to compute $H(J\|r)^{-1} \pmod{p''q''}$, $T_2^{H(J\|r)^{-1}} = g^{H(J\|r)e_2 H(J\|r)^{-1}} = g^{e_2} = \Omega \pmod{n_T}$ ……**(e3)**

Searching for (A_2, A_3) in withdrawal database, if $A_3/A_2^{K_1} = \Omega \pmod{n_T}$, T finds the coin owner. $W_2 = (J, r, T_2,$ **(e3)**), and (W, W_1, W_2) prove that C with PK_C is the coin owner.

✧ Unconditional coin-tracing from T

T gets A_2 and A_3 from withdrawal. Since $A_2 = g^{e_1} \pmod{n_T}$, $A_3 = g^{e_2}h^{e_1} \pmod{n_T}$, $g^{e_2} = A_3 / A_2^{K_1} \pmod{n_T}$. Then in payment protocol, $T_2 = g^{H(J\|r)e_2} = (A_3/A_2^{K_1})^{H(J\|r)} \pmod{n_T}$.

So the coin from the wallet will be identified when it is spent. $W_3 = SPK\ (\alpha: g^{e_2} = A_3/A_2^{\alpha} \wedge h = g^{\alpha} \wedge T_2 = (A_3/A_2^{\alpha})^{H(J\|r)})$, and ($W_1$, T_2, W_3) prove that the e-coins are from the traced wallet.

Lost-coin tracing

✧ Lost-coin tracing from B

After Loss Register, all coins in the lost e-wallet will be traced if $T_{2x} = g^{e_2} \pmod{n_T}$ is published in system. That is to say, $T_2 = g^{H(J\|r)e_2} = (T_{2x})^{H(J\|r)} \pmod{n_T}$ in payment.

So the coin from the lost wallet will be identified when it is spent. And (W_{loss}, W, W_1) prove that the spending coin has been registered for lost-coin tracing by the actual owner with PK_C.

7. Security of the Proposed Scheme

As we analyzed in section 6.4, the main idea of the proposed scheme is to construct the special knowledge proof. To achieve it, replacing random r with $(J+e_2)^{-1} \pmod{n_s}$ to construct the special challenge-response $s_\sigma = (J+e_2)^{-1} \pmod{n_s} - c(e_1 - 2^{\lambda_1})$. Therefore, the key issues of security are:

- the security of construction of zero-knowledge proof is preserved; (Theorem 1)
- $(J+e_2)^{-1} \pmod{n_s}$ can be used as a pseudorandom function in this construction; (Theorem 2)
- the special knowledge proof has the zero-knowledge property. (Theorem 3)

Then we provide the proofs of the key issues first.

Theorem 1. *When each serial number of coin Θ is used only once, none of proven parameters can be computed from* ***challenge-response*** *equations under discrete logarithm assumption.*

Proof . In the payment protocol, PK_3 proves that Θ is generated correctly, and PK_3 and PK_4 guarantee that one **challenge-response** of PK_1 is $s_\sigma = (J+e_2)^{-1} \pmod{n_s} - c(e_1 - 2^{\lambda_1})$ ……**(e4)**. So using a Θ only once indicate that there is only one equation set **(E1)** for each Θ, **(e4)** is the only one that is not a standard **challenge-response**. Because the other knowledge proofs are the standard knowledge proofs, the related proven parameters cannot be computed from challenge-response equation set. In addition, the single **(e4)** has arbitrary solutions in their value ranges of e_1 and e_2. When K coins from the same wallet are spent, **(e4)** from the respective equation sets **(E1)** constitute **(E2)**, and the coefficient matrix M_1 of **(E2)** is as follows:

$$\textbf{(E2)}\ \begin{cases} (J_1+e_2)^{-1} \pmod{n_s} - c_1 e_1 = s_{\sigma 1} - c_1 2^{\lambda_1} \\ (J_2+e_2)^{-1} \pmod{n_s} - c_2 e_1 = s_{\sigma 2} - c_2 2^{\lambda_1} \\ \vdots \\ (J_{K-1}+e_2)^{-1} \pmod{n_s} - c_{K-1} e_1 = s_{\sigma(K-1)} - c_{K-1} 2^{\lambda_1} \\ (J_K+e_2)^{-1} \pmod{n_s} - c_K e_1 = s_{\sigma K} - c_K 2^{\lambda_1} \end{cases}$$

$$(J_1+e_2)^{-1} \quad (J_2+e_2)^{-1} \ \ldots \ (J_{K-1}+e_2)^{-1} \quad (J_K+e_2)^{-1} \quad e_1$$

$$\begin{pmatrix} 1 & 0 & \dots & 0 & 0 & c_1 \\ 0 & 1 & \dots & 0 & 0 & c_2 \\ & & \vdots & & & \\ 0 & 0 & \dots & 1 & 0 & c_{K-1} \\ 0 & 0 & \dots & 0 & 1 & c_K \end{pmatrix}$$

The row vectors of M_1 are:

$\alpha_1 = (1\ 0\ \dots\ 0\ 0\ c_1)$ $\qquad$ $\alpha_{K-1} = (0\ 0\ \dots\ 1\ 0\ c_{K-1})$

$\alpha_2 = (0\ 1\ \dots\ 0\ 0\ c_2)$ ..... $\alpha_K = (0\ 0\ \dots\ 0\ 1\ c_K)$

If want to compute $(J_1+e_2)^{-1}$, it indicates that there is the matrix as below (the constant $\Delta \neq 0$) after elementary transformation, but it will be proven to be infeasible.

$$\begin{matrix} (J_1+e_2)^{-1} & (J_2+e_2)^{-1} \dots & (J_{K-1}+e_2)^{-1} & (J_K+e_2)^{-1} & e_1 \end{matrix}$$
$$\begin{pmatrix} \Delta & 0 & \dots & 0 & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 & c_2 \\ & & \vdots & & & \\ 0 & 0 & \dots & 1 & 0 & c_{K-1} \\ 0 & 0 & \dots & 0 & 1 & c_K \end{pmatrix}$$

So there are $(k_1,k_2,\dots,k_{K-1},k_K)$ satisfying $(\Delta\ 0 \dots 0\ 0\ 0) = k_1\cdot\alpha_1+k_2\cdot\alpha_2+\dots+k_{K-1}\cdot\alpha_{K-1}+k_K\cdot\alpha_K$.

That is to say,

$$\begin{cases} k_1 + 0 + \dots + 0 + 0 = \Delta \\ 0 + k_2 + \dots + 0 + 0 = 0 \\ \vdots \\ 0 + 0 + \dots + k_{K-1} + 0 = 0 \\ 0 + 0 + \dots + 0 + k_K = 0 \\ k_1c_1 + k_2c_2 + \dots + k_{K-1}c_{K-1} + k_Kc_K = 0 \end{cases}$$

We have $k_2 = k_3 = \dots = k_{K-1} = k_K = 0$, $k_1 = \Delta \neq 0$, so $c_1 = 0$. However, it will be found by the customer when spending the coin if $c=0$. Therefore, computing $(J_1+e_2)^{-1} \pmod{n_s}$ is infeasible. And for the same reason, $(J_i+e_2)^{-1} \pmod{n_s}$ can not be figured out, where $2 \le i \le K$.

Anyone who tries to compute e_1 must get the following coefficient matrix ($\Delta \neq 0$).

$$\begin{matrix} (J_1+e_2)^{-1} & (J_2+e_2)^{-1} \dots & (J_{K-1}+e_2)^{-1} & (J_K+e_2)^{-1} & e_1 \end{matrix}$$
$$\begin{pmatrix} 0 & 0 & \dots & 0 & 0 & \Delta \\ 0 & 1 & \dots & 0 & 0 & c_2 \\ & & \vdots & & & \\ 0 & 0 & \dots & 1 & 0 & c_{K-1} \\ 0 & 0 & \dots & 0 & 1 & c_K \end{pmatrix}$$

So there are $(k_1,k_2,\dots,k_{K-1},k_K)$ satisfying $(0\ 0 \dots 0\ 0\ \Delta) = k_1\cdot\alpha_1+k_2\cdot\alpha_2+\dots+k_{K-1}\cdot\alpha_{K-1}+k_K\cdot\alpha_K$.

That is to say,

$$\begin{cases} k_1 + 0 + \dots + 0 + 0 = 0 \\ 0 + k_2 + \dots + 0 + 0 = 0 \\ \vdots \\ 0 + 0 + \dots + k_{K-1} + 0 = 0 \\ 0 + 0 + \dots + 0 + k_K = 0 \\ k_1c_1 + k_2c_2 + \dots + k_{K-1}c_{K-1} + k_Kc_K = \Delta \end{cases}$$

We have $k_1 = k_2 = \dots = k_{K-1} = k_K = 0$, and it contradicts $\Delta \neq 0$, so figuring out e_1 is also infeasible.

So e_1 and e_2 cannot be computed from **(E1)** or **(E2)**. Therefore, none of proven parameters can be computed if the customer never spends the same coin twice. □

Theorem 2. *Suppose $(s(k), 2^{a(k)}, \varepsilon(k))$ -DDHI assumption holds.* (i) *$f_{1e_2}(J)=a^{(J+e_2)^{-1}}$ is the $(s'(k), \varepsilon'(k))$ pseudo-random function for the one who has no knowledge of e_2, and $s'(k) = s(k)/[2^{a(k)}\cdot poly(k)]$, $\varepsilon'(k) = 2^{a(k)}\cdot \varepsilon(k)$.* (ii) *If the one who has no knowledge of e_2 cannot use the extended Euclidean algorithm to compute inverse of $f_{2e_2}(J)$, $f_{2e_2}(J)=(J+e_2)^{-1} \pmod{n_s}$ is the $(s''(k), \varepsilon''(k))$ pseudorandom function, and $s''(k) = s(k)/[2^{a(k)}\cdot poly(k)]$, $\varepsilon''(k) = 2^{a(k)}\cdot \varepsilon(k)$.*

Proof . For the sake of contradiction, suppose there exists the algorithm $\mathcal{A}$, which ($s'(k)$, q, $\varepsilon'(k)$)-breaks the pseudorandom function, i.e., runs in $s'(k)$ time, and can identify $f_{1e_2}(J_0) = a'^{(J_0+e_2)^{-1}}$ with the probability at least $1/2+\varepsilon'(k)$, where $(J_0, a'^{(J_0+e_2)^{-1}})$ is any unseen point of the function $f_{1e_2}(J)$. Then we can construct the algorithm $\mathcal{B}$, which interacts with $\mathcal{A}$ to break the q-DDHI assumption —— based on a random instance $(a, a^{\alpha}, \ldots, a^{(\alpha^q)})$ of the q-DHI problem, the goal of algorithm $\mathcal{B}$ is to identify $a^{1/\alpha}$. Let $J_0 = \alpha - e_2$ where e_2 and α are unknown to $\mathcal{A}$ and $\mathcal{B}$. And the error probability could be decreased by changing J_0 and executing this algorithm sufficiently many times. Note that $(a, a^{e_2}, \ldots, a^{(e_2^q)})$ can be computed from $(a, a^{\alpha}, \ldots, a^{(\alpha^q)})$ according to Binomial Theorem.

Input to the reduction: $(a, a^{\alpha}, \ldots, a^{(\alpha^q)}, \Gamma) \in G^{q+2}$, where Γ is either $a^{1/\alpha}$ or a random element in G. Its goal is to output 1 if $\Gamma = a^{1/\alpha}$ and 0 otherwise.

Then $\mathcal{B}$ interacts with $\mathcal{A}$ as follows:

Query: $\mathcal{A}$ outputs a list of distinct q_s serial numbers $J_1, \ldots, J_{qs}$, and $q_s < q$. Because $\mathcal{A}$ reveals the queries in advance, we could assume $\mathcal{A}$ outputs q-1 serial numbers to be responded (if the number is less, reduce the value of q to satisfy $q=q_s+1$).

Response: $\mathcal{B}$ computes the polynomial $f(y)=\prod_{i=1}^{q-1}(y+J_i)$.

Then we have $f(y) = \sum_{i=0}^{q-1} \alpha_i y^i$ by expanding $f(y)$, where $\alpha_0, \ldots, \alpha_{q-1} \in Z_n$ are the coefficients of the polynomial $f(y)$.

$$a' = \prod_{i=0}^{q-1} a^{\alpha_i e_2^{\,i}} = a^{f(e_2)} \qquad \prod_{i=1}^{q} a^{\alpha_{i-1} e_2^{\,i}} = a^{e_2 f(e_2)} = (a')^{e_2}$$

Then give a' to $\mathcal{A}$. Ans we can assume that $f(e_2) \neq 0$, because otherwise, $e_2 = -J_i$ for some i means $\mathcal{B}$ has got the secret e_2. Then for each $i = 1, \ldots, q$-1, $\mathcal{B}$ computes R_i and responds it to $\mathcal{A}$ as follows: let $f_i(y)$ be the polynomial $f_i(y) = f(y)/(y + J_i) = \prod_{j=1, j\neq i}(y + J_i)$. We can expand it as $f_i(y) = \sum_{j=0}^{q-2} \beta_j y^j$.

$$R_i = \prod_{j=0}^{q-2} a^{\beta_j e_2^{\,j}} = a^{f_i(e_2)} = (a')^{1/(e_2 + J_i)}$$

So $\mathcal{B}$ can give $\mathcal{A}$ the q-1 responses $R_1, \ldots, R_{q-1}$ without the knowledge of e_2.

Challenge: $\mathcal{A}$ claims that he can distinguish $a'^{1/(e_2+J_*)} = a^{f(e_2)/(e_2+J_*)}$ from a random element in G. If $J_* \neq J_0$, then repeats the algorithm again. Otherwise, $\mathcal{A}$ claims that he can distinguish $a'^{1/(e_2+J_0)} = a^{f(e_2)/(e_2+J_0)}$ from a random element in G where $J_0 \notin \{J_1, \ldots, J_{q-1}\}$. Now compute

$$f(y) / (y + J_0) = \sum_{i=0}^{q-2} \gamma_i y^i + \gamma_{-1}/(y + J_0)$$

where $\gamma_{-1} \neq 0$ since $f(y) = \prod_{i=1}^{q-1}(y+J_i)$ and $J_0 \notin \{J_1, \ldots, J_{q-1}\}$. So

$$a'^{1/(e_2+J_0)} = a^{f(e_2)/(e_2+J_0)} = a^{\sum_{i=0}^{q-2} \gamma_i e_2^{\,i} + \gamma_{-1}/(e_2 + J_0)} = \Gamma_0 \cdot a^{\gamma_{-1}/(e_2 + J_0)} = \Gamma_0 \cdot (a^{1/\alpha})^{\gamma_{-1}}$$

Because $\Gamma_0 = \prod_{i=0}^{q-2}(a^{e_2^{\,i}})^{\gamma_i}$ is computed from this algorithm, if $\Delta=a'^{1/(e_2+J_0)}$ and $\Gamma=a^{1/\alpha}$, $\Delta=\Gamma_0 \cdot \Gamma^{\gamma_{-1}}$. So $\mathcal{B}$ can distinguish $a^{1/\alpha}$ from $(a, a^{\alpha}, \ldots, a^{\alpha^q})$ according to the guess about $\Delta \stackrel{?}{=} a'^{1/(e_2+J_0)}$ from $\mathcal{A}$.

Guess: $\mathcal{A}$ outputs a guess $b \in \{0, 1\}$ for Δ, and then $\mathcal{B}$ outputs a guess $b' \in \{0, 1\}$ for Γ.

The main running time of the reduction is the time of simulating oracle queries. Since $\mathcal{A}$ can make at most $s'(k)$ queries, the running time of $\mathcal{B}$ is $s'(k) \cdot [2^{a(k)} \cdot poly(k)]$. And the advantage of $\mathcal{B}$ is $\varepsilon'(k)/2^{a(k)}$. So $s'(k) = s(k)/[2^{a(k)} \cdot poly(k)]$, and $\varepsilon'(k) = 2^{a(k)} \cdot \varepsilon(k)$.

Challenge and Guess following $\mathcal{C}$: Algorithm $\mathcal{C}$, which ($s''(k)$, $\varepsilon''(k)$)-breaks the *pseudo-randomness* of $f_{2'e_2}(J)=(J+e_2)^{-1}$, claims to be able to distinguish $P^*=(J_0+e_2)^{-1}$ from a random element without the knowledge of e_2 (Theorem 1 guarantees the extended Euclidean algorithm cannot be used). If $\Gamma = a^{1/\alpha}$, $a'^{P^*}=a'^{1/(e_2+J_0)} = \Gamma_0 \cdot \Gamma^{\gamma_{-1}} \ldots\ldots$**(e5)**. Giving P^* to algorithm $\mathcal{C}$, $\mathcal{C}$ outputs a guess $d_{\text{guess}} \in \{0, 1\}$ for $P^* \stackrel{?}{=} (J_0+e_2)^{-1}$. Then $\mathcal{B}$ outputs d'_{guess} for Γ according to **(e5)**.

Therefore, without the knowledge of e_2, the pseudorandom property of $f_{2}'_{e_2}(J)=(J+e_2)^{-1}$ is proven if no one can use extended Euclidean algorithm to compute inverse of it, so the pseudorandom property of $f_{2e_2}(J)=(J+e_2)^{-1} \pmod{n_s}$ is proven. Likewise, $f_{2e_2}(J)=(J+e_2)^{-1} \pmod{n_s}$ is a $(s''(k), \varepsilon''(k))$ pseudo-random function, and $s''(k)=s(k)/[2^{a(k)}\cdot poly(k)]$ and $\varepsilon''(k)= 2^{a(k)}\cdot \varepsilon(k)$.□

Theorem 3. *If customer never spends the same coin twice, the knowledge proof in payment protocol has the statistical zero-knowledge property under q-DDHI assumption.*

Proof . We only need to prove that $(c, s_\sigma)\in\{0,1\}^k\times\{-2^{k+\lambda_2}+2^{\lambda_2}+1,\ldots, n_s+2^{k+\lambda_2}-2^{\lambda_2}-1\}$ from the non-standard knowledge proof has the statistical zero- knowledge property (it will be proven in Theorem 6 that the special knowledge proof is the knowledge proof of e_1). To be concise, suppose $y=a_1^{e_1}$, $(c, s_\sigma)\in \{0,1\}^k\times\{-2^{k+\lambda_2}+2^{\lambda_2}+1,\ldots, n_s+2^{k+\lambda_2}-2^{\lambda_2}-1\}$ satisfying $c = H(y^c\cdot a_1^{s_\sigma-c2^{\lambda_1}})$.

According to Theorem 2, $(J_i+e_2)^{-1}$ in the construction of payment is a random element to the one who has no knowledge of e_2. The generating mode in the withdrawal protocol guarantees that $(J_i+e_2) \pmod{n_s}$ uniformly distributes over $[1, n_s-1]$, and so does $(J_i+e_2)^{-1} \pmod{n_s}$, since every inverse of $(J_i+e_2) \pmod{n_s}$ is unique and distinct from each other in $[1, n_s-1]$.

To prove statistical zero-knowledge property of the knowledge proof, let us show that the simulator which uniformly chooses the challenge, can simulate this protocol-conversation which is statistically indistinguishable from the protocol-conversation with C.

The simulator randomly chooses $\bar{c}$ from $\{0,1\}^k$ and $\bar{s}_\sigma$ from $\{-2^{k+\lambda_2}+2^{\lambda_2}+1,\ldots, n_s+2^{k+\lambda_2}-2^{\lambda_2}-1\}$ satisfying uniform distribution. Using the values, the simulator computes $\Delta = y^{\bar{c}}a_1^{\bar{s}_\sigma-\bar{c}2^{\lambda_1}} \pmod{n}$ [40]. For proving that the values are statistical indistinguishable from a view of a protocol run with C, we will show the probability distribution $P_{S_\sigma}(s_\sigma)$ of response s_σ from C and the probability distribution $P_{\bar{S}_\sigma}(\bar{s}_\sigma)$.

In payment protocol, $s_\sigma = (J+e_2)^{-1} \pmod{n_s}-c(e_1-2^{\lambda_1})$, where $0<(J+e_2)^{-1}<n_s$, and $(J+e_2)^{-1}$ uniformly distributes over $[1, n_s-1]$ as we analyze above. The e_1 is chosen from $[2^{\lambda_1}-2^{\lambda_2}, 2^{\lambda_1}+2^{\lambda_2}]$, and c (i.e., the output of $H(\ )$) can be any distribution over $\{0,1\}^k$. And $l_{p_s}= \varepsilon(\lambda_2+k)+2k+2$.

$$P_{S_\sigma}(s_\sigma)\begin{cases} = 0 & \text{for} \quad s_\sigma < -2^{k+\lambda_2}+2^{\lambda_2}+1 \\ \le (n_s-1)^{-1} & \text{for} \quad -2^{k+\lambda_2}+2^{\lambda_2}+1 \le s_\sigma < 2^{k+\lambda_2}-2^{\lambda_2}+1 \\ = (n_s-1)^{-1} & \text{for} \quad 2^{k+\lambda_2}-2^{\lambda_2}+1 \le s_\sigma \le n_s-2^{k+\lambda_2}+2^{\lambda_2}-1 \\ \le (n_s-1)^{-1} & \text{for} \quad n_s-2^{k+\lambda_2}+2^{\lambda_2}-1 < s_\sigma \le n_s+2^{k+\lambda_2}-2^{\lambda_2}-1 \\ = 0 & \text{for} \quad n_s+2^{k+\lambda_2}-2^{\lambda_2}-1< s_\sigma \end{cases}$$

Let us provide a brief explanation of $P_{S_\sigma}(s_\sigma)$.

$P(s_\sigma=2^{k+\lambda_2}-2^{\lambda_2}+1) = \sum P((J+e_2)^{-1} \pmod{n_s}-c(e_1-2^{\lambda_1}) = 2^{k+\lambda_2}-2^{\lambda_2}+1)$

$= P((J+e_2)^{-1} \pmod{n_s}=1, c(e_1-2^{\lambda_1})=-2^{k+\lambda_2}+2^{\lambda_2}) + P((J+e_2)^{-1} \pmod{n_s}=2, c(e_1-2^{\lambda_1}) =-2^{k+\lambda_2}+2^{\lambda_2}+1)$

$\quad + \ldots + P((J+e_2)^{-1} \pmod{n_s}= n_s-1, c(e_1-2^{\lambda_1})= n_s-2^{k+\lambda_2}+2^{\lambda_2}-2)$

$= (n_s-1)^{-1} [P(c(e_1-2^{\lambda_1})=-2^{k+\lambda_2}+2^{\lambda_2}) + P(c(e_1-2^{\lambda_1}) =-2^{k+\lambda_2}+2^{\lambda_2}+1) + \ldots + P(c(e_1-2^{\lambda_1})= n_s-2^{k+\lambda_2}+2^{\lambda_2}-2)]$

$= (n_s-1)^{-1} [P(c(e_1-2^{\lambda_1})=-2^{k+\lambda_2}+2^{\lambda_2}) + P(c(e_1-2^{\lambda_1}) =-2^{k+\lambda_2}+2^{\lambda_2}+1) + \ldots + P(c(e_1-2^{\lambda_1})= 2^{k+\lambda_2}-2^{\lambda_2})]$

$= (n_s-1)^{-1} \cdot 1 = (n_s-1)^{-1}$

The **Fig. 6** presents the distribution of $P_{\bar{S}_\sigma}(\bar{s}_\sigma)$ and $P_{S_\sigma}(s_\sigma)$.

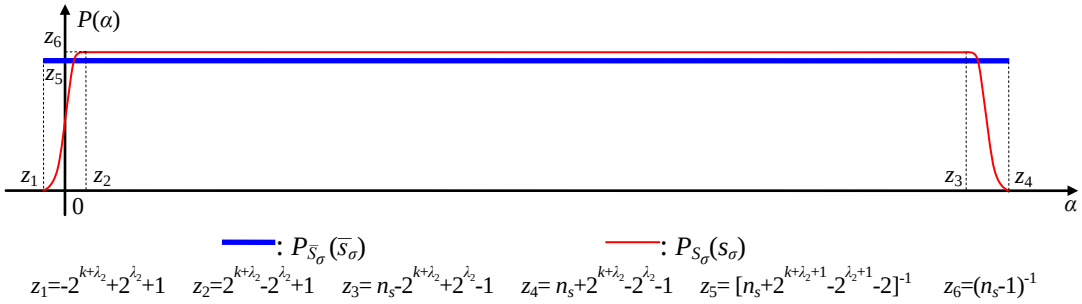


Fig. 6. The Distribution of $P_{\bar{S}\sigma}(\bar{s}_\sigma)$ and $P_{S\sigma}(s_\sigma)$

Then we have

$$\sum_{\alpha\in Z}|P_S(\alpha)-P_{\bar{S}}(\bar{\alpha})| = \sum_{\alpha\in\{-2^{k+\lambda_2}+2^{\lambda_2}+1,\ldots,\, n_s+2^{k+\lambda_2}-2^{\lambda_2}-1\}}|P_S(\alpha)-[n_s+2^{k+\lambda_2+1}-2^{\lambda_2+1}-2]^{-1}|$$
$$< (z_6-z_5)(z_3-z_2+1)+z_6\cdot[(z_4-z_3)+(z_2-z_1)]$$
$$= \{(n_s-1)^{-1}-[n_s+2^{k+\lambda_2+1}-2^{\lambda_2+1}-2]^{-1}\}\cdot(n_s-2^{k+\lambda_2+1}+2^{\lambda_2+1}-2)+(n_s-1)^{-1}[(2^{k+\lambda_2+1}-2^{\lambda_2+1})+(2^{k+\lambda_2+1}-2^{\lambda_2+1})]$$
$$< (n_s-1)^{-1}\cdot(2^{k+\lambda_2+1}-2^{\lambda_2+1}-1)+(n_s-1)^{-1}(2^{k+\lambda_2+2}-2^{\lambda_2+2})$$
$$= (n_s-1)^{-1}\cdot(2^{k+\lambda_2+1}-2^{\lambda_2+1}-1+2^{k+\lambda_2+2}-2^{\lambda_2+2}) < (n_s-1)^{-1}\cdot(2^{k+\lambda_2+3}-2^{\lambda_2+3}) < (n_s-1)^{-1}\cdot 2^{k+\lambda_2+3} < 1/2^{(\varepsilon-1)(\lambda_2+k)+2k-2}$$

For $\varepsilon>1$, the denominator of last term of the above computation is over a polynomial in input length, so the distributions of s_σ and $\bar{s}_\sigma$ are statistical indistinguishable. Therefore, according to the Definition 2, the interactive protocol of payment is the honest-verifier statistical zero-knowledge proof. □

Theorem 4. *Under Discrete Logarithm assumption, the* ***non-standard*** *challenge-response* $s_\sigma=(J+e_2)^{-1} \pmod{n_s}-c(e_1-2^{\lambda_1})$ *is computed correctly in the challenge-response equation set* **(E1)**.

Proof . In payment protocol, $PK_3(e_2: \Theta=a_{1J}^{(J+e_2)^{-1} \pmod{n_s}})$ and $PK_4(\delta: \Theta=a_{1J}^{\delta} \wedge d_1=T_1^{r_1}/(a^{r_2}a_1^{\delta}a_2^{r_3}h^{r_4}))$ prove that $d_1=T_1^{r_1}/[a^{r_2}a_1^{(J+e_2)^{-1}(\mathrm{mod}\ n_s)}a_2^{r_3}h^{r_4}]$, which is used as a commitment of zero-knowledge proof about the discrete logarithm knowledge of $a_0=T_1^{e}/(a^{x}a_1^{e_1}a_2^{e_2}h^{ew})$, C shows s_σ satisfying

$$d_1 = a_0^{c}T_1^{s_1-c2^{\gamma_1}}/(a^{s_2-c2^{\lambda_1}}a_1^{s_\sigma-c2^{\lambda_1}}a_2^{s_3}h^{s_4}) \pmod{n} = T_1^{r_1}/[a^{r_2}a_1^{(J+e_2)^{-1}(\mathrm{mod}\ n_s)}a_2^{r_3}h^{r_4}]$$

Without the discrete logarithm knowledge of (a_0,T_1,a,a_1,a_2,h) to each other, the exponents of a_1 are equal: $s_\sigma+c(e_1-2^{\lambda_1}) = \underline{(J+e_2)^{-1} \pmod{n_s}} + k_0\cdot n_s$……**(e6)**, where $\underline{(J+e_2)^{-1} \pmod{n_s}}$ denotes the value of the inverse of $(J+e_2)$ is in $[1, n_s-1]$ and k_0 is any integer. If $k_0=0$, $s_\sigma=\underline{(J+e_2)^{-1} \pmod{n_s}}-c(e_1-2^{\lambda_1})$ so that the tracing can be performed correctly. We present how to guarantee $k_0=0$.

(1) If C executes payment protocol honestly, $k_0=0$. From $c\in[1,2^k-1]$, $(e_1-2^{\lambda_1})\in[-2^{\lambda_2},2^{\lambda_2}]$, and $\underline{(J+e_2)^{-1} \pmod{n_s}}\in[1, n_s-1]$, the probability of $s_\sigma\in[-2^{k+\lambda_2}+2^{\lambda_2}+1, n_s+2^{k+\lambda_2}-2^{\lambda_2}-1]$ is 1. And S verifies whether $s_\sigma\in[-2^{k+\lambda_2}+2^{\lambda_2}+1, n_s+2^{k+\lambda_2}-2^{\lambda_2}-1]$ or not in payment protocol.

(2) If C is dishonest and computes **(e6)** choosing $k_0 \neq 0$

○ If C computes **(e6)** choosing $k_0 = 1$, i.e., $s_\sigma=\underline{(J+e_2)^{-1} \pmod{n_s}}+n_s-c(e_1-2^{\lambda_1})$. In this case, if $s_\sigma\in[-2^{k+\lambda_2}+2^{\lambda_2}+1, n_s+2^{k+\lambda_2}-2^{\lambda_2}-1]$, it means $-n_s-2^{k+\lambda_2}+2^{\lambda_2}+1 \le \underline{(J+e_2)^{-1} \pmod{n_s}}-c(e_1-2^{\lambda_1}) \le 2^{k+\lambda_2}-2^{\lambda_2}-1$. We use $Prob_{k_0=1}$ to denote the probability that S cannot find the deceit,

$$Prob_{k_0=1} = Prob\{\ -n_s-2^{k+\lambda_2}+2^{\lambda_2}+1 \le \underline{(J+e_2)^{-1} \pmod{n_s}}-c(e_1-2^{\lambda_1}) \le 2^{k+\lambda_2}-2^{\lambda_2}-1\ \}$$
$$= Prob\{\ -2^{k+\lambda_2}+2^{\lambda_2}+1 \le \underline{(J+e_2)^{-1} \pmod{n_s}}-c(e_1-2^{\lambda_1}) \le 2^{k+\lambda_2}-2^{\lambda_2}-1\ \}$$
$$= Prob\{\ 1 \le \underline{(J+e_2)^{-1} \pmod{n_s}} \le c(e_1-2^{\lambda_1})+2^{k+\lambda_2}-2^{\lambda_2}-1\ \}$$

$\underline{(J+e_2)^{-1} \pmod{n_s}}$ uniformly distributes over $[1, n_s-1]$ since $(J+e_2) \pmod{n_s}$ uniformly distributes over $[1, n_s-1]$ according to the withdrawal protocol, $e_1\in[2^{\lambda_1}-2^{\varepsilon(\lambda_2+k)}, 2^{\lambda_1}+2^{\varepsilon(\lambda_2+k)}]$ according to W.

Then we have $(e_1-2^{\lambda_1})\in[-2^{\varepsilon(\lambda_2+k)}, 2^{\varepsilon(\lambda_2+k)}]$. Therefore,

$Pro_{k_0=1} < \{(2^k-1)\cdot 2^{\varepsilon(\lambda_2+k)}+2^{k+\lambda_2}-2^{\lambda_2}-1\}/(n_s-1) < 2^{\varepsilon(\lambda_2+k)+k+1}/(n_s-1)$

Because n_s is a l_{ps}-bits prime, and $l_{p_s} = \varepsilon(\lambda 2+k)+2k+2$,

$Pro_{k_0=1} < 2^{\varepsilon(\lambda_2+k)+k+1}/(n_s-1) < 1/2^k$, and it is negligible (usually, bit-length of hash function k =128 or 160).

○ If C computes **(e6)** choosing $k_0 = -1$, i.e., $s_\sigma=(J+e_2)^{-1} \pmod{n_s}-n_s-c(e_1-2^{\lambda_1})$. In this case, if $s_\sigma\in[-2^{k+\lambda_2}+2^{\lambda_2}+1, n_s+2^{k+\lambda_2}-2^{\lambda_2}-1]$, it means $n_s-2^{k+\lambda_2}+2^{\lambda_2}+1 \le (J+e_2)^{-1} \pmod{n_s} -c(e_1-2^{\lambda_1}) \le 2n_s+2^{k+\lambda_2}-2^{\lambda_2}-1$.

$$\begin{aligned} Prob_{k_0=-1} &= Prob\{\, c(e_1-2^{\lambda_1})+ n_s-2^{k+\lambda_2}+2^{\lambda_2}+1 \le (J+e_2)^{-1} \pmod{n_s} \le n_s-1 \,\} \\ &< Prob\{\, (2^k-1)\cdot-2^{\varepsilon(\lambda_2+k)}+n_s-2^{k+\lambda_2}+2^{\lambda_2}+1 \le (J+e_2)^{-1} \pmod{n_s} \le n_s-1 \,\} \\ &< [2^{\varepsilon(\lambda_2+k)+k}+2^{k+\lambda_2}-2^{\lambda_2}-2]/(n_s-1) < 2^{\varepsilon(\lambda_2+k)+k+1}/(n_s-1) < 1/2^k \end{aligned}$$

○ If C computes **(e6)** with $|k_0|>1$, it is easy to see that $s_\sigma \notin [-2^{k+\lambda_2}+2^{\lambda_2}+1, n_s+2^{k+\lambda_2}-2^{\lambda_2}-1]$.

As above, $Prob_{|k_0|=1}$ is negligible and $Prob_{|k_0|>1}= 0$. Therefore, C has to computes **(e6)** choosing $k_0=0$, i.e., it guarantees that $s_\sigma=(J+e_2)^{-1} \pmod{n_s}-c(e_1-2^{\lambda_1})$ is computed correctly. □

The proposed scheme is based on our previous work [30]. Considering the similar proofs of Theorem 5-Theorem 8 had been provided in [30], we just present Theorem 5-Theorem 8.

Theorem 5. *Under S-RSA assumption, any PPT adversary except* B, *can not, with non-negligible probability, computes* $(e_1,e_2,x,[A,e],a_0)$ *s.t.* $A^e=a_0a^xa_1^{e_1}a_2^{e_2} \pmod n$ *with* $e_1,x\in\Lambda$, $e_2\in[0, 2n_s-2]$ *and* $e\in\Gamma$ *which is different from wallets generated in withdrawal protocol.*

Theorem 6. *The knowledge proof in the proposed payment protocol is the proof of C's knowledge of his wallet parameters* $(e_1,e_2,x,[A,e])$ *under S-RSA assumption.*

Theorem 7. *No PPT adversary except* C *can, with non-negligible probability, generate the spending proof that is not actually generated by* C, *but the spending proof is proven to be generated by* C *under the Discrete Logarithm assumption.*

Theorem 8. *Our compact E-cash scheme with (BSetup, TSetup, CSetup, Withdraw, Spend, Deposit, UnconditionallyTrace, LossCoinTrace, DoubleSpendTrace) guarantees Balance, Complete-tracing, Anonymity of customer, Strong Exculpability under S-RSA assumption and q-DDHI assumption in random oracle model.*

8. Efficiency Analysis

8.1 Storage Space of some E-cash Systems

To compare clearly, **Table 2** presents the storage space of each stage in some E-cash systems. For achieving comparable secure level, the bit-length of order of cyclic group G is 1024 [17, 16, 8, 20, 19, 41, 26, 30] and our scheme, and the prime order p of G_1 and G_2 in bilinear map is 160 bits in [8, 21, 22, 31, 23]. We select L=10 in [22, 26] and select l=10 in [8, 21, 23] and our scheme accordingly, which make these schemes provide the similar functions.

Table 2. Storage space for 1 coin in E-cash schemes

	[17]	[16]	[8]		[21]	[20]	[19]	[41]	[22]	[26]	[23]	[31]	[30]	Our scheme
			System 1	System 2										
Withdrawal [bit]	4416	4296	5632	6112 +3lx	1714	11570	9408	6784	24800	5120	38720	2720 +320·2^n	5940	8406
Payment [bit]	7732	7604	15565	(21l+12) x	4768	9276	4864	6836	4800	1033216	19840	7680	2996	7836
Deposit [bit]	7732	7604	18022	(21l+12) x	5280	9276	4864	6836	4800	1033216	19840	7680	2996	7836
Space complexity for 2^n coins	$O(2^n)$	$O(2^n)$	$O(n)$	$O(n)$	$O(n)$	$O(2^n)$	$O(2^n)$	$O(2^n)$	$O(2^n)$	$O(n)$	$O(n)$	$O(2^n)$	$O(2^n)$	$O(n)$

l: quantity of bits of coin counter x: is 160 in bilinear group, or 1024 in RSA group

The space complexity for 2^n coins is $O(n)$ in compact E-cash, that is to say, in [8, 21, 23] and our scheme, the storage space for 2^n coins is the same as it (shown in **Table 2**) for 1 coins.

8.2 Computation Cost of some E-cash Systems

Since multi-based exponentiations and bilinear pairings are the main computations of the protocols in the systems, they are presented in **Table 3**, while the slight computations, such as modular addition computations and hash computations, are all neglected.

Table 3. Computation cost for 1 coin in E-cash schemes

		[17]	[16]	[8] System 1	[8] System 2	[21]	[20]	[19]	[41]	[22]	[26]	[23]	[31]	[30]	Our scheme
Withdrawal	$\mathbf{F}_1$	18	12	12	$8k$+10	9	24	25	44	4299	8	242	$17+2^{2n}$	21	18
	$\mathbf{F}_2$	0	0	0	$3k$	2	0	0	0	44	0	294	7	0	0
Payment	$\mathbf{F}_1$	20	18	40	$72l$+58	37	7	13	29	35	3520+x_1	50	42	8	17
	$\mathbf{F}_2$	0	0	0	0	8	0	0	0	14	0	246	36	0	0
Deposit	$\mathbf{F}_1$	7	6	11	$21l$+17	10	6	5	6	13	3520+x_2	50	13	3	6
	$\mathbf{F}_2$	0	0	0	0	4	0	0	0	8	0	246	18	0	0
Computation complexity of withdrawing 2^n coins		$O(2^n)$	$O(2^n)$	$O(1)$	$O(k)$	$O(1)$	$O(2^n)$	$O(2^n)$	$O(2^n)$	$O(2^n)$	$O(1)$	$O(1)$	$O(2^{2n})$	$O(2^n)$	$O(1)$

$\mathbf{F}_1$: multi-based exponentiation $\mathbf{F}_2$: bilinear pairing l: bit-length of coin counter
k: cheating probability is 2^{-k} at most in the system2 of [8] x_1, x_2: related to system parameters

In compact E-cash system, to withdraw 2^n coins, the user performs the withdrawal protocol only once, so [8, 21, 23] and our scheme achieve better efficiency for 2^n coins. Note that in divisible e-cash, the user can withdraw the coin with the value of 2^n coins and use $O(n)$ space to store it, but for spending one coin with $1/2^n$ of total value, the preparation work is costly.

8.3 Our solution to two problems

Our main work is to solve two problems, one is achieving the complete and practical tracing, and the other one is solving the efficiency problem caused by tracing customer's coins if he double-spends a coin. **Table 4** presents our solution to the practical and complete tracing, that is to say, it presents the available tracing functions in the E-cash schemes.

Table 4. The available tracings in the E-cash systems

	[17]	[16]	[8] System 1	[8] System 2	[21]	[20]	[19]	[41]	[22]	[26]	[23]	[31]	[30]	Our scheme
Double-spender tracing from Bank	*N/A*	*N/A*	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**
Double-spender's coin tracing from Bank	*N/A*	*N/A*	*N/A*	**Yes**	*N/A*	*N/A*	*N/A*	*N/A*	*N/A*	*N/A*	*N/A*	**Yes**	*N/A*	**Yes**
Unconditional owner tracing from TTP	**Yes**	**Yes**	*N/A*	*N/A*	*N/A*	*N/A*	*N/A*	**Yes**	*N/A*	*N/A*	*N/A*	*N/A*	**Yes**	**Yes**
Unconditional coin tracing from TTP	**Yes**	*N/A*	*N/A*	*N/A*	*N/A*	*N/A*	*N/A*	**Yes**	*N/A*	*N/A*	*N/A*	*N/A*	**Yes**	**Yes**
Lost-coin tracing from Bank	*N/A*	*N/A*	*N/A*	*N/A*	*N/A*	*N/A*	*N/A*	*N/A*	*N/A*	*N/A*	*N/A*	*N/A*	**Yes**	**Yes**
Compact E-cash which can trace double-spender's coins				✓								✓		✓

Yes: the scheme provides this tracing ***N/A***: this tracing function is not available

According to **Table 4**, the system 2 of [8], [31] and ours are compact E-cash systems which can trace double-spender's coins (providing it without TTP), and to the best of our knowledge, they include all compact E-cash systems providing this function. Then **Table 5** presents the

solution to the efficiency problem caused by this tracing. And for achieving the comparable secure level, l=10, k=100, x=1024 (the meanings of the parameters are in **Table 2** and **Table 3**).

Table 5. Compact E-cash with double-spender's coin-tracing

		Storage space [bit]			Computation cost					
					F_1			F_2		
		[8] System 2	[31]	Our scheme	[8] System 2	[31]	Our scheme	[8] System 2	[31]	Our scheme
For 2^n coins	Withdrawal	36832	330400	8406	810	1048593	18	300	7	0
For 1 coin	Payment	227328	7680	7836	778	42	17	0	36	0
	Deposit	227328	7680	7836	227	13	6	0	18	0

F_1: multi-based exponentiation F_2: bilinear pairing

9. Conclusion

Anonymity is good, but it could be abused for crimes or cause trouble when E-cash is lost. Complete tracing can solve this problem. However, it also threatens the honest user's privacy. The reasonable solution is to separate different tracing functions provided by different entities and choose the available ones according to the circumstances. Practical tracing can achieve it. To achieve the practical and complete tracing, another serious problem must be solved, i.e., how to trace double-spender's coins efficiently. For solving it, we propose the particular knowledge proof, and using it honestly keeps perfect zero-knowledge property, while using it dishonestly leaks the information of proven knowledge. Since it changes the inner construction of standard zero-knowledge proof, we provide the complete proofs of it. Consequently, the practical and complete tracing is also efficient.

References

[1] Y. Chen, J. S. Chou, H. M. Sun, and M. H. Cho, "A novel electronic cash system with trustee-based anonymity revocation from pairing," *Electronic Commerce Research and Applications*. vol.10, no.6, pp. 673-682, 2011. Article (CrossRef Link)

[2] Z. Tan, "An Off-line Electronic Cash Scheme Based on Proxy Blind Signature," *The Computer Journal*, vol. 54, no. 4, pp. 505-512, 2011. Article (CrossRef Link)

[3] D. Chaum, "Blind signatures for untraceable payments," in *Proc. of CRYPTO'82*, pp. 199–203, 1983. Article (CrossRef Link)

[4] Pin-Chang Su and Chien-Hua Tsai, "New Proxy Blind Signcryption Scheme for Secure Multiple Digital Messages Transmission Based on Elliptic Curve Cryptography," *KSII Transactions on Internet and Information Systems*, vol. 11, no. 11, pp. 5537-5555, 2017. Article (CrossRef Link)

[5] Md. Abdullah Al Rahat Kutubi, Kazi Md. Rokibul Alam, Rafaf Tahsin, G. G. Md. Nawaz Ali, Peter Han Joo Chong and Yasuhiko Morimoto, "An Offline Electronic Payment System Based on an Untraceable Blind Signature Scheme," *KSII Transactions on Internet and Information Systems*, vol. 11, no. 5, pp. 2628-2645, 2017. Article (CrossRef Link)

[6] Zhen Zhao, Jie Chen, Yueyu Zhang and Lanjun Dang, "An Efficient Revocable Group Signature Scheme in Vehicular Ad Hoc Networks," *KSII Transactions on Internet and Information Systems*, vol. 9, no. 10, pp. 4250-4267, 2015. Article (CrossRef Link)

[7] Run Xie, Chunxiang Xu, Chanlian He and Xiaojun Zhang, "An Efficient Dynamic Group Signature with Non-frameability," *KSII Transactions on Internet and Information Systems*, vol. 10, no. 5, pp. 2407-2426, 2016. Article (CrossRef Link)

[8] J. Camenisch, S. Hohenberger, and A. Lysyanskaya, "Compact e-cash," in *Proc. of Advances in Cryptology– EUROCRYPT 2005*, pp. 302-321, 2005. Article (CrossRef Link)

[9] S. von Solms and D. Naccache, "On blind signatures and perfect crimes," *Computers & Security*, vol. 11, pp.581-583, 1992. Article (CrossRef Link)

[10] E. Brickell, P. Gemmell, and D. Kravitz, "Trustee-based tracing extensions to anonymous cash and the making of anonymous change," in *Proc. of 6th annual ACM-SIAM symposium on Discrete algorithms*, pp. 457-466 ,1995. Article (CrossRef Link)

[11] M. Stadler, J. Piveteau, and J. Camenisch, "Fair blind signatures," in *Proc. of Advances in Cryptology Eurocrypt'95*, pp. 209-219, 1995. Article (CrossRef Link)

[12] A. Lysyanskaya and Z. Ramzan, "Group blind digital signatures: A scalable solution to electronic cash," in *Proc. of FC'98*, pp. 184-197, 1998. Article (CrossRef Link)

[13] G. Maitland and C. Boyd, "Fair electronic cash based on a group signature scheme," *Information and Communications Security*, pp. 461-465, 2001. Article (CrossRef Link)

[14] H. Oros and C. Popescu, "A Secure and Efficient Off-line Electronic Payment System for Wireless Networks," *Intl. J. of Computers, Comm. and Control*, Suppl. Issue Vol. V, No. 4, pp. 551-557, 2010. Article (CrossRef Link)

[15] J. Zhang, L. Ma, and Y. Wang, "Fair E-Cash System without Trustees for Multiple Banks," in *Proc. of CISW 2007*, pp. 585-587, 2007. Article (CrossRef Link)

[16] S. Canard, C. Delerablée, A. Gouget, E. Hufschmitt, F. Laguillaumie, H. Sibert, J. Traoré, and D. Vergnaud, "Fair E-Cash: Be Compact, Spend Faster," in *Proc. of ISC 2009: Information Security*, pp. 294-309, 2009. Article (CrossRef Link)

[17] S. Canard and J. Traoré, "On fair e-cash systems based on group signature schemes," in *Proc. of ACISP2003*, pp. 237-248, 2003. Article (CrossRef Link)

[18] W. Qiu, K. Chen"A new offline privacy protecting e-cash system with revokable anonymity," *Information Security*, pp.177, 2002. Article (CrossRef Link)

[19] H. Wang, J. Cao, and Y. Zhang, "A flexible payment scheme and its role-based access control," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 3, pp. 425-436, 2005. Article (CrossRef Link)

[20] J. Liu, P. Tsang, and D. Wong, "Recoverable and untraceable e-cash," in *Proc. of PKI*, pp. 206-214, 2005. Article (CrossRef Link)

[21] M. Au, W. Susilo, and Y. Mu, "Practical compact e-cash," in *Proc. of the 12th Australasian conference on Information security and privacy 2007*, pp. 431-445, 2007. Article (CrossRef Link)

[22] M. Au, W. Susilo, and Y. Mu, "Practical anonymous divisible e-cash from bounded accumulators," *Financial Cryptography and Data Security*, pp. 287-301, 2008. Article (CrossRef Link)

[23] M. Belenkiy, M. Chase, M. Kohlweiss, and A. Lysyanskaya, "Compact e-cash and simulatable VRFs revisited," in *Proc. of Pairing-Based Cryptography–Pairing 2009*, pp. 114-131, 2009. Article (CrossRef Link)

[24] S. Brands, "Untraceable off-line cash in wallet with observers," in *Proc. of CRYPTO'93*, pp. 302-318, 1993. Article (CrossRef Link)

[25] S. Brands and C. v. W. e. Informatica, "An efficient off-line electronic cash system based on the representation problem," *CWI Technical Report CS-R9323*, Citeseer, 1970. Article (CrossRef Link)

[26] S. Canard and A. Gouget, "Divisible e-cash systems can be truly anonymous," in *Proc. of Advances in Cryptology-EUROCRYPT 2007*, pp. 482-497, 2007. Article (CrossRef Link)

[27] Z. Eslami and M. Talebi, "A new untraceable off-line electronic cash system," *Electronic Commerce Research and Applications*, vol. 10, no. 1, pp. 59–66, 2011. Article (CrossRef Link)

[28] Schoenmakers, B., "Security aspects of the E-cash™ payment system," *State of the Art in Applied Cryptography*, pp. 338–352, 1998. Article (CrossRef Link)

[29] W. S. Juang, "RO-cash: An efficient and practical recoverable pre-paid offline e-cash scheme using bilinear pairings," *Journal of Systems and Software*, vol. 83, pp. 638-645, 2010. Article (CrossRef Link)

[30] B. Lian, G. L. Chen and J. H. Li, "Provably secure E-cash system with practical and efficient complete tracing," *International Journal of Information Security*, vol. 13, no. 3, pp. 271-289, Apr. 2014. Article (CrossRef Link)

[31] M. Au, Q Wu, W Susilo, Y Mu, "Compact E-Cash from Bounded Accumulator," in *Proc. of CT-RSA'07*. LNCS, vol. 4377, pp. 178-195, 2007. Article (CrossRef Link)

[32] B. Lian, G. Chen, M. Ma, J. Li, "Periodic K-Times Anonymous Authentication with Efficient Revocation of Violator's Credential," *IEEE Transactions on Information, Forensics and Security*, VOL. 10, NO. 3, pp. 543-557. 2015. Article (CrossRef Link)

[33] E. Fujisaki and T. Okamoto, "Statistical zero knowledge protocols to prove modular polynomial relations," in *Proc. of Advances in Cryptology—CRYPTO'97*, pp. 16-30, 1997. Article (CrossRef Link)

[34] D. Boneh, "The decision diffie-hellman problem," *Algorithmic Number Theory*, pp. 48-63, 1998. Article (CrossRef Link)

[35] G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik, "A practical and provably secure coalition-resistant group signature scheme," in *Proc. of Advances in Cryptology—CRYPTO 2000*, pp. 255-270, 2000. Article (CrossRef Link)

[36] J. Camenisch, "Group signature schemes and payment systems based on the discrete logarithm problem," *PhD thesis*, vol. 2 of ETH Series in Information Security an Cryptography, Hartung-Gorre Verlag, Konstanz. ISBN 3-89649-286-1, 1998. Article (CrossRef Link)

[37] Y. Dodis and A. Yampolskiy, "A Verifiable Random Function With Short Proofs and Keys," in *Proc. of Public Key Cryptography*, vol. 3386 of LNCS, pp. 416-431, 2005. Article (CrossRef Link)

[38] J. Camenisch and Anna Lysyanskaya, "A signature scheme with efficient protocols," in *Proc. of Security in Communication Networks'02*, vol. 2576 of LNCS, pages 268–289. 2002. Article (CrossRef Link)

[39] J. Camenisch and M. Michels, "Proving in zero-knowledge that a number is the product of two safe primes," in *Proc. of Advances in Cryptology—EUROCRYPT'99*, pp. 107-122, 1999. Article (CrossRef Link)

[40] C.P. Schnorr, "Efficient Signature Generation by Smart Cards," *Journal of Cryptology*, vol 4, pp. 161-174, 1991. Article (CrossRef Link)

[41] B. Lian, GL. Chen, JH. Li, " A Provably Secure and Practical Fair E-cash Scheme," in *Proc. of 2010 IEEE International Conference on Information Theory and Information Security*, 2010. Article (CrossRef Link)

Bin Lian received the M.S. degree in cryptography from Southwest Jiaotong University in 2005 and the Ph.D. degree in cryptography from Shanghai Jiao Tong University in 2015. He is with the Ningbo Institute of Technology, Zhejiang University, Ningbo, China. His research interests include cryptography, cryptographic protocol and technology of network security.

Gongliang Chen received his B.S. in Peking University, and M.S. degree in Chinese Academy of Science. In 1993, he received his Ph.D. degree in Université de Saint Etienne, France. He is also a visiting scholar of Université Paris VI, France. He is currently a professor at the School of Information Security Engineering, Shanghai Jiao Tong University, Shanghai. His main research area includes cryptographic theory and technology of network security.

Jialin Cui received the M.S. degree from Zhejiang University in 2005. He is with the Ningbo Institute of Technology, Zhejiang University, Ningbo, China. He is currently pursuing the Ph.D. degree with Ningbo University. His research interests include cryptography, security protocol and cryptographic applications.

Dake He received his M.S. degree in Xidian University, Xian in 1981. He is a professor and doctoral supervisor at Southwest Jiaotong University, Chengdu. He is also the director of national high performance computing center (Chengdu). His research area includes cryptography, communications system security, security engineering of information system, parallel computing and applied mathematics.