Provably Secure Off-Line E-Cash Scheme with Efficient Anonymity Revoking

Chun-I Fan
Department of Computer Science and Engineering
National Sun Yat-sen University
Email: cifan@faculty.nsysu.edu.tw

Vincent Shi-Ming Huang
Department of Computer Science and Engineering
National Sun Yat-sen University
Email: vincent.sm.huang@gmail.com

Yao-Chun Yu
Department of Computer Science and Engineering
National Sun Yat-sen University
Email: kisskevin524@gmail.com

***Abstract*—Due to the fast progress of the internet technologies, electronic commerce becomes more and more popular. Many people and businesses deal with their transactions via the internet. The technologies of credit cards, electronic tickets, electronic cash (e-cash), and other advanced services have realized the vision of electronic commerce. In this manuscript, we propose an off-line e-cash scheme with anonymity, unlinkability, double-spending checking, and anonymity control. In an off-line e-cash scheme, the bank or the third party (TTP) must be able to revoke the anonymity of a user who doubly spent her/his e-cash(s). In our proposed e-cash scheme, the bank can fast derive the identity of the user who doubly spent her/his e-cash(s) without the participation of TTP. If some illegal transactions are reported, TTP can also directly revoke the anonymity of the user who spent her/his e-cash(s) in the illegal transactions. In addition, the police need to trace a specific user in some situations. We also propose a mechanism to achieve this goal, call traceability.**

***Index Terms*—Electronic Cash, Double Spending, Unlinkability, Anonymity, Chameleon Hash Functions**

I. Introduction

The widespread networks make the electronic commerce more and more popular. Lots of businesses utilize computers and networks to deal with the transactions of their commercial activities. Besides, mobile devices, say cellphones, have more storage and convenient network connection interfaces, like Bluetooth, WiFi, and 3G networks. In electronic commerce, a user can trade by using the mobile device everywhere. Hence, an electronic payment mechanism is necessary for electronic commerce. In this manuscript, we propose an e-cash scheme which can be used in electronic commerce. E-cash resembles the paper cash in the real world where a payer owns anonymity and unlinkability. In 1982, David Chaum [1] proposed an e-cash scheme by exploiting his RSA-based blind signature. In Chaum's e-cash scheme, a user withdraws an e-cash and pays it without revealing her/his identity. The bank cannot collude with shops to trace any user's consuming behavior. However, e-cash is easily duplicated. Since e-cash cannot be spent doubly, the bank needs to cope with the double-spending problem. Once the bank detects an e-cash which has been doubly spent, it must be able to find out the owner. Besides, in some e-cash schemes, the bank or the trusted third party (TTP) can revoke the anonymity of a user when necessary. In general, e-cash can be classified into two types which are on-line e-cash [2] [3] [1] [4] and off-line e-cash [5] [6] [7] [8]. In an on-line e-cash scheme, when a shop receives an e-cash, the shop will send the e-cash to the bank to make double-spending checking immediately after verifying the correctness of the e-cash. In an off-line e-cash scheme, when a shop receives an e-cash, the shop will store the e-cash if it is correct and send it to the bank to perform double-spending checking after a period of time. The bank can prevent double-spending in an on-line e-cash scheme because it performs double-spending checking before accepting an e-cash. However, in an off-line e-cash scheme, the bank cannot prevent double-spending in advance. Therefore, the bank must be able to revoke the anonymity of the user who doubly spent her/his e-cash.

In our proposed off-line e-cash scheme, each user possesses anonymity and unlinkability when spending e-cash(s). If a user doubly spends her/his e-cash, the bank can detect it and efficiently derive the identity of the user without any help of TTP.

Besides, if an e-cash has been spent in an illegal transaction and reported to TTP, TTP can revoke the anonymity of the owner of the e-cash. Our e-cash scheme also allows the police to trace a specific user. Consequently, the proposed off-line e-cash scheme is with anonymity, unlinkability, and anonymous control which contains revokeability and traceability.

A. Organization of the Manuscript

The rest of this manuscript is organized as follows. Section II briefly reviews the related works about electronic cash schemes. In Section III, we describe the architecture of the proposed scheme and the requirements which are necessary in our electronic cash scheme, and then we present our scheme in Section IV and in Section V. In Section VI, we make characteristic analysis and compare our scheme with the others. The formal security proofs for the proposed scheme are shown in Section VII. Finally, a concluding remark is given in Section VIII.

II. Preliminary

A. David Chaum's Blind Signature

In 1983, Chaum proposed a blind signature scheme [1] based on the RSA cryptosystem [9]. Chaum's blind signature scheme contains *KeyGen*, *Blinding*, *Signing*, *Unblinding*, and *Verifying* algorithms. The details of the blind signature scheme are described as follows.

- *KeyGen*: The input is a security parameter 1^k. The algorithm will output the public and private keys (e, n) and d.
- *Blinding*: A user randomly selects $a \in \mathbb{Z}_n^*$ and computes $\alpha = a^e H(m) \bmod n$ where H is a one-way hash function and m is a message. Then the user sends the signer α.
- *Signing*: The signer computes $t = \alpha^d \bmod n$ and sends t back to the user.
- *Unblinding*: After receiving t, the user computes $s = ta^{-1} \bmod n$ and then gets a signature s on m.
- *Verifying*: Any one can verify whether s is a valid signature on m by checking if $s^e \equiv H(m) \pmod{n}$ is true or not.

B. Krawczyk and Rabin's Scheme (Chameleon Signatures)

In 2000, H. Krawczyk and T. Rabin proposed a hash function, called *trapdoor hash function* [10]. They used it to construct *chameleon signatures*. A chameleon hash function is associated with a public key HK and a private key TK. If one knows HK, he can compute the associated hash function. But without the private key, it is infeasible to find two inputs which are mapped to the same output. If anyone who has the private key, he can easily find collisions for every given input. A trapdoor hash function is a probabilistic function h_{HK} such that it is hard to find a collision when only HK is given, but it is easy to find the collision when TK is also given. Formally speaking, given only HK, it is difficult to find two messages m_1, m_2 and two auxiliary numbers r_1, r_2, such that $h_{HK}(m_1, r_1) = h_{HK}(m_2, r_2)$, but given (HK, TK) and m_1, m_2, r_1, it is easy to find r_2 such that $h_{HK}(m_1, r_1) = h_{HK}(m_2, r_2)$.

Definition 1: A trapdoor hash family consists of a pair $(\mathcal{I}, \mathcal{H})$ such that:

- $\mathcal{I}$ is a probabilistic polynomial time key generation algorithm that on input 1^k outputs a pair (HK, TK), such that the sizes of HK and TK are polynomially related to k.
- $\mathcal{H}$ is a family of randomized hash functions. Every hash function in H is associated with a hash key HK, and is applied to a message from a space M and a random element from a finite space R. The output of the hash function h_{HK} does not depend on TK.

A trapdoor hash family $(\mathcal{I}, \mathcal{H})$ has the following properties:

1) Collision resistance: There is no efficient (probabilistic polynomial time) algorithm that on input public key HK can find two pairs $(m_1, r_1), (m_2, r_2) \in M \times R$ where $m_1 \neq m_2$ such that $h_{HK}(m_1, r_1) = h_{HK}(m_2, r_2)$.
2) Trapdoor collisions: There is an efficient (probabilistic polynomial time) algorithm that given $(HK, TK) \leftarrow \mathcal{I}(1^k)$, a pair $(m_1, r_1) \in M \times R$, and an additional message $m_2 \in M$, finds a value $r_2 \in R$ such that $h_{HK}(m_1, r_1) = h_{HK}(m_2, r_2)$.
3) Uniform Probabilistic Distribution: If $r_1 \in R$ is distributed uniformly, $m_1 \in M$, and

$(m_2, r_2) \in M \times R$ such that $h_{HK}(m_1, r_1) = h_{HK}(m_2, r_2)$, then r_2 is computationally indistinguishable from uniform in R.

The Construction of the Chameleon Hashing Based on Discrete Log Assumption

- The key generation algorithm $\mathcal{I}$: Randomly choose a safe prime number $p \in \{0,1\}^k$ such that $p = 2q + 1$ where q is a large enough prime number, and select a generater $g \in \mathbb{Z}_p^*$ of order q. Choose a random integer $x \in \mathbb{Z}_q^*$ and compute $y = g^x \bmod p$. Output the public hash key (p, q, g, y) and the private trapdoor key x.
- The hash family $\mathcal{H}$: Given $HK = (p, q, g, y)$ and a message m, $h_{HK} : \mathbb{Z}_q \times \mathbb{Z}_q \to \mathbb{Z}_p^*$ is defined as $h_{HK}(m, r) = g^m y^r \bmod p$ where $r \in \mathbb{Z}_q^*$ is a random-selected integer.

III. The Architecture and Requirements

In this section, we describe the architecture of our proposed off-line e-cash scheme and discuss the requirements for an off-line e-cash system.

A. The architecture of our e-cash scheme

There are four entities, **Bank**, **Shop**, **Customer**, and **Judge** in our e-cash scheme. The architecture is shown in Fig. 1. First, the judge issues a judge device to the bank. The bank embeds the judge device into its system. When a customer wants to withdraw her/his e-cash(s), the judge device will be a participant in our withdrawal protocol. When a customer wants to make a payment to a shop, the customer performs an offline e-cash payment protocol with the shop. The shop stores the received e-cash after verifying the e-cash. Finally, the shop deposits the received e-cash(s) into the bank after a period of time, and the bank will deal with the double-spending checking on the e-cash(s). If double-spending happens, the bank will retrieve the identity from the e-cash directly.

B. Requirements

In an offline e-cash system, there are some requirements which must be satisfied.

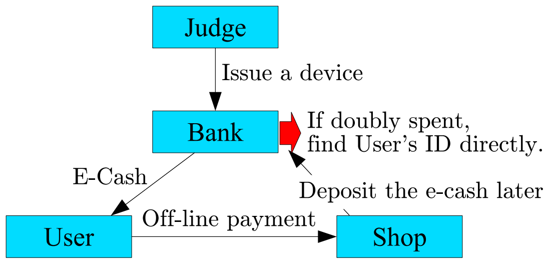

Fig. 1. The architecture of our scheme

1) Anonymity and Unlinkability:
Anonymity and Unlinkability are the basic requirements for every e-cash system. Anonymity means that when an e-cash is shown, none can know who withdrew this e-cash. Unlinkability means that both of the bank and shops cannot trace a user's consuming behavior. Due to anonymity and unlinkability, the user can preserve her/his privacy in an e-cash scheme.
2) Unforgeability:
None can generate an e-cash except the bank. It is infeasible for anyone to forge an e-cash without the bank's private key.
3) Double-Spending Resistance:
When the bank receives an offline e-cash, it is able to check whether the received e-cash was doubly spent or not. The bank will store all of the spent e-cash(s) in a database. When a user doubly spends her/his e-cash, the bank can detect it via checking the database.
4) Anonymity Control:
This consists of Revokability and Traceability. Revokability contains criminal revoking and double-spending revoking. In criminal revoking, the judge is able to revoke the anonymity of a given e-cash which has been spent in some illegal transactions. In double-spending revoking, the bank or the judge can find the identity who doubly spent an e-cash. Traceability means that the bank can trace a specific user if necessary. In our scheme, if the police want to trace some user, it will ask the bank to perform the tracing procedure.
5) Tamper Resistance:
None can tamper the information in an e-cash. In our scheme, when a user withdraws an e-cash from the bank, her/his identity will be embedded in the e-cash. This property guarantees that the embedded identity cannot be tampered.

6) No Swindling:
None can spend the e-cash except the real owner of the e-cash. In our scheme, after a user withdraws an e-cash, she/he will get a secret which will be used in the payment protocol. Therefore, the e-cash can be spent by the owner only.

IV. Our Proposed Scheme

In our scheme, there are two main protocols which are **Withdrawal Protocol** and **Payment Protocol**, and four entities **User**, **Bank**, **Shop** and **Judge**. We use Chaum's blind signature [1] and chameleon hash functions [10] to design our e-cash scheme. A user withdraws an e-cash from the bank by running the Withdrawal Protocol and spends her/his e-cash by performing the Payment Protocol. Before describing our protocols, we define and explain some notations as follows.

A. Notations

- E_x: This is a semantically secure encryption function where x can be a symmetric key or a public key. Let k_{A_B} denote a shared key between entity A and entity B and pk_C be a public key of entity C. E_{A_B} is a symmetric encryption function and E_{pk_C} is an asymmetric encryption function.
- H: This is a one-way and collision free hash function. It is computationally infeasible to derive the input message from a given hashed value. And it is impossible to find two different input messages with the same hashed value.
- h_{HK}: This is a chameleon hash function which was proposed by Krawczyk and Rabin and described in Section II-B.
- $(pk\_j,\ sk\_j)$: This is the judge's public-private key.
- l_k, l_r: These are security parameters.
  - l_k is the bit length of a session key.
  - l_r is the bit length of a random string.
- A judge device: The judge device is issued by the judge. It will be integrated into the system of the bank. Before the judge issues the device, it can perform a public-testing to show that the device is fair. Any information embedded in the hardware device cannot be modified by anyone else. We can make use of the technique of TPM (Trusted Platform Module) [11] to implement the judge device. The judge device contains
  - a random number/string generator
  - a public-key encryption/decryption function
  - a symmetric-key encryption/decryption function
  - a public-private key (pk_j, sk_j) of the judge
  - a public key of the bank
  - two hash functions, H and h_{HK}

B. Initialization

First, the bank selects two distinct large primes (p_b, q_b) at random and computes $n_b = p_b q_b$. Then it also randomly selects an integer e_b such that $GCD(\phi(n_b), e_b) = 1$ and $1 < e_b < \phi(n_b)$ and computes d_b such that $e_b d_b \equiv 1 \pmod{\phi(n_b)}$ where $\phi(n_b) = (p_b - 1)(q_b - 1)$. It randomly chooses a safe prime p and an element $g \in \mathbb{Z}_p^*$ of order q where $p = kq + 1$, q is a prime and k is an integer. Finally, It also selects a one-way hash function H. Then the bank publishes (n_b, e_b, p, q, g, H) where (p, q, g) is the public parameter of the chameleon hash function.

The judge generates its public-private key (pk_j, sk_j). Then it publishes its public key and embeds and $(pk\_j, sk\_j, H, h_{Hk}, n_b, e_b)$ into a temper-resistant device which will be issued to the bank.

C. The Withdrawal Protocol

1) The Normal Withdrawal Protocol: When a user wants to withdraw an e-cash from her/his account, she/he has to be authenticated by the bank and then runs the withdrawal protocol. In this manuscript, we will not discuss the authentication mechanism which can be any secure authentication protocol for the bank to authenticate the user. We depict the withdrawal protocol in Fig. 2 and describe it as follows.

1) User $\rightarrow$ Bank: $E_{pk\_j}(k, m, r)$
The user randomly chooses three strings (k, m, r) where $k \in \{0,1\}^{l_k}$ and $m, r \in \mathbb{Z}_q^*$. Then she/he computes $E_{pk\_j}(k, m, r)$ and sends $E_{pk\_j}(k, m, r)$ to the bank.
2) Bank $\rightarrow$ The judge device: $(E_{pk\_j}(k, m, r), \mu)$
After the user is authenticated by the bank, the

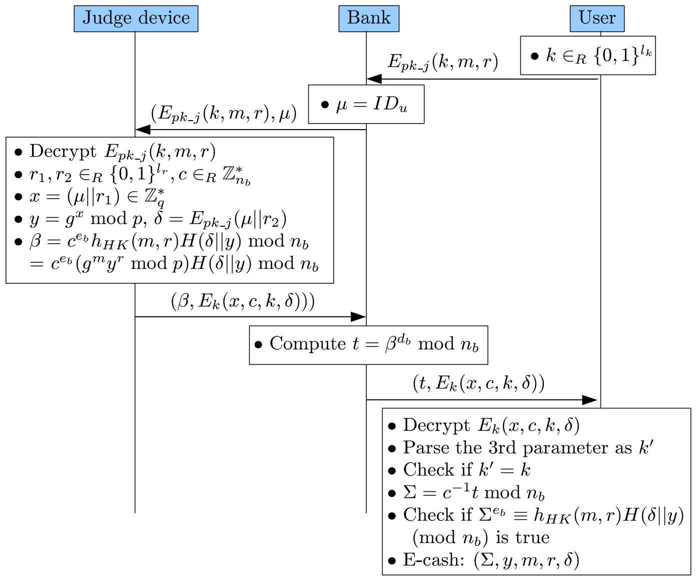
Judge device
Bank
User
• $k \in_R \{0,1\}^{l_k}$
$E_{pk\_j}(k,m,r)$
• $\mu = ID_u$
$(E_{pk\_j}(k,m,r),\mu)$
• Decrypt $E_{pk\_j}(k,m,r)$
• $r_1, r_2 \in_R \{0,1\}^{l_r}, c \in_R \mathbb{Z}^*_{n_b}$
• $x = (\mu||r_1) \in \mathbb{Z}^*_q$
• $y = g^x \bmod p$, $\delta = E_{pk\_j}(\mu||r_2)$
• $\beta = c^{e_b} h_{HK}(m,r)H(\delta||y) \bmod n_b$
$= c^{e_b}(g^m y^r \bmod p)H(\delta||y) \bmod n_b$
$(\beta, E_k(x,c,k,\delta)))$
• Compute $t = \beta^{d_b} \bmod n_b$
$(t, E_k(x,c,k,\delta))$
• Decrypt $E_k(x,c,k,\delta)$
• Parse the 3rd parameter as k'
• Check if $k' = k$
• $\Sigma = c^{-1}t \bmod n_b$
• Check if $\Sigma^{e_b} \equiv h_{HK}(m,r)H(\delta||y)$
$\pmod{n_b}$ is true
• E-cash: $(\Sigma, y, m, r, \delta)$

Fig. 2. The Withdrawal Protocol

bank knows the identity ID_u of the user. The bank sets $\mu = ID_u$ and inputs $E_{pk\_j}(k,m,r)$ and μ into the judge device.

3) The judge device $\rightarrow$ Bank: $(\epsilon, E_k(x,c,k,\delta))$
After receiving $E_{pk\_j}(k,m,r)$ and μ, the judge device uses sk_j to decrypt $E_{pk\_j}(k,m,r)$ and gets (k,m,r). Then it randomly chooses three strings (r_1, r_2, c) where $r_1, r_2 \in \{0,1\}^{l_r}$ and $c \in \mathbb{Z}^*_{n_b}$. Then it computes $x = (\mu \| r_1) \in \mathbb{Z}^*_q$, $\delta = E_{pk\_j}(\mu \| r_2)$, and $y = g^x \bmod p$. Finally, it computes $\beta = c^{e_b}(g^m y^r \bmod p)H(\delta||y) \bmod n_b = c^{e_b} h_{HK}(m,r)H(\delta||y) \bmod n_b$ and outputs $(\beta, E_k(x,c,k,\delta))$ to the bank where $HK = (p,q,g,y)$.

4) Bank $\rightarrow$ User: $(t, E_k(x,c,k,\delta))$
After receiving $(\beta, E_k(x,c,k,\delta))$ from the judge device, the bank computes $t = \beta^{d_b} \bmod n_b$ and returns $(t, E_k(x,c,k,\delta)$ to the user.

5) Unblinding:
After receiving $(t, E_k(x,c,k,\delta))$, the user decrypts $E_k(x,c,k,\delta)$ and parses the 3rd parameter in the decryption result as k'. Then she/he checks whether $k' = k$. If true, she/he computes $\Sigma = c^{-1}t \bmod n_b$. Finally, the user obtains an e-cash $(\Sigma, y, m, r, \delta)$ and she/he can check whether the signature is true or not by examining if $\Sigma^{e_b} \equiv h_{HK}(m,r)H(\delta||y) \pmod{n_b} \equiv (g^m y^r \bmod p)H(\delta||y) \pmod{n_b}$. If it is invalid, the user will notify the bank and the bank will retransmit $(t, E_k(x,c,k,\delta))$ to the user.

D. The Payment Protocol

In Fig. 3, the payment protocol contains four steps.

1) Shop $\rightarrow$ User: (m')
When a user wants to make a payment with a shop, the shop will randomly choose a string r_s and compute $m' = (ID_s \| r_s)$ such that $m' \in \mathbb{Z}^*_q$ where ID_s is the shop's identity. Then the shop sends m' to the user.

2) User $\rightarrow$ Shop: (Σ, r', y, δ)
After receiving m', the user computes

$$r' = x^{-1}(m + xr - m') \bmod q. \qquad (1)$$

Then, she/he sends (Σ, r', y, δ) to the shop.

3) Shop $\rightarrow$ Bank: $(\Sigma, y, m', r', \delta)$
After receiving (Σ, y, r', δ), the shop verifies if $\Sigma^{e_b} \equiv h_{HK}(m', r')H(\delta||y) \pmod{n_b}$ where $HK = (p,q,g,y)$. If true, the shop accepts the e-cash and stores $(\Sigma, y, m', r', \delta)$. The shop will send the received e-cash to the bank later.

4) Bank: Acceptance or Rejection
After a period of time, the shop deposits $(\Sigma, y, m', r', \delta)$ to the bank. The bank first verifies the e-cash by checking if $\Sigma^{e_b} \equiv h_{HK}(m', r')H(\delta||y) \pmod{n_b}$ and (Σ, y, δ) has not existed in the database. If both of them are true, the bank stores $(\Sigma,\ y,\ m',\ r',\ \delta)$ in the database and deposits the e-cash into the shop's account.

V. Anonymity Control

Anonymity control contains Revokability and Traceability. Revokability makes it possible for the bank or the judge to revoke the anonymity of the owner of an e-cash which was doubly spent or spent in an illegal transaction. When the bank receives an e-cash, it must check whether the e-cash is doubly spent or not. In our scheme, the bank can easily detect double-spending and directly find out the user who doubly spent her/his e-cash. When there is an e-cash reported from an illegal transaction, the judge

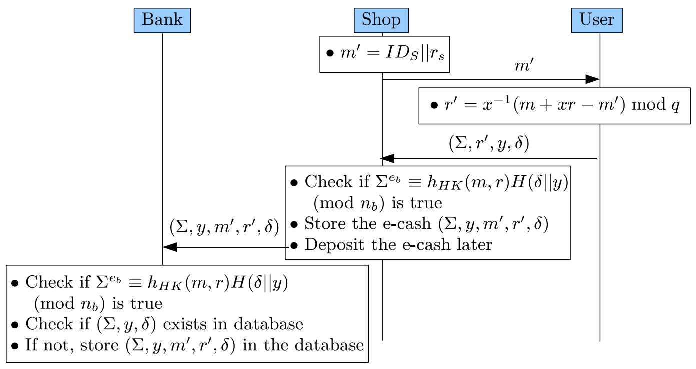
Bank
Shop
User
• $m' = ID_S||r_s$
m'
• $r' = x^{-1}(m + xr - m') \bmod q$
(Σ, r', y, δ)
• Check if $\Sigma^{e_b} \equiv h_{HK}(m,r)H(\delta||y) \pmod{n_b}$ is true
• Store the e-cash $(\Sigma, y, m', r', \delta)$
• Deposit the e-cash later
$(\Sigma, y, m', r', \delta)$
• Check if $\Sigma^{e_b} \equiv h_{HK}(m,r)H(\delta||y) \pmod{n_b}$ is true
• Check if (Σ, y, δ) exists in database
• If not, store $(\Sigma, y, m', r', \delta)$ in the database

Fig. 3. The Payment Protocol.

can revoke the owner's identity of the reported e-cash. For the traceability, the police can ask the bank to trace some e-cash(s) which were withdrawn by some specific users if necessary.

A. *Revokability*

- On double-spending:
  If someone doubly spent her/his e-cash, the bank can revoke the anonymity of the owner of the e-cash. When a shop sends an e-cash $(\Sigma_2, y_2, m_2, r_2, \delta_2)$ to the bank, the bank will check whether the received e-cash has existed in the database. If the bank can find another e-cash $(\Sigma_1, y_1, m_1, r_1, \delta_1)$ where $\Sigma_1 = \Sigma_2$, $y_1 = y_2$, and $\delta_1 = \delta_2$, the bank will obtain $m_1 + xr_1 \equiv m_2 + xr_2 \pmod q$. Then the bank can compute
  $$x = \frac{m_1 - m_2}{r_2 - r_1} \bmod q. \qquad (2)$$
  where x is formatted as $(ID_u||r')$ where ID_u is the identity of the user of the e-cash and r' is a random string. Therefore, the bank can efficiently derive the identity of the user without the judge's help.
- On illegal transaction:
  Sometimes, if a user spent an e-cash for some illegal transactions such as money laundering, the e-cash will be reported to the judge and the privacy of the user will be revoked. The judge can decrypt $\delta = E_{pk\_j}(\mu||r_2)$ by its private key sk_j and get μ where μ is the identity of the user.

B. *Traceability*

In some situation, the police may want to trace a specific user. Assume that the police's public key and private key are (e_p, n_p) and (d_p, p_p, q_p) based on the RSA cryptosystem. If the user's identity is ID_u, the police will compute $s_P = H(ID_u)^{d_p} \bmod n_p$ and send (ID_u, s_P) to the bank. Once the user performs the withdrawal protocol with the bank, the bank will input $(E_{pk\_j}(k, m, r), \mu, s_p)$ to the judge device. The judge device will verify whether $s_p^{e_p} \equiv H(\mu) \pmod{n_p}$ or not. If true, it will return $(\beta, E_k(x, c, k, \delta), k^{e_p} \bmod n_p)$ to the bank. Then the bank will forward $(E_k(x, c, k, \delta), k^{e_p} \bmod n_p)$ to the police. After receiving $(E_k(x, c, k, \delta), k^{e_p} \bmod n_p)$, the police can get k and decrypt $E_k(x, c, k, \delta)$. Then it puts δ in a blacklist and sends the blacklist to all shops. When the user ID_u spends her/his e-cash $(\Sigma, h, m, r, \delta)$ in a shop, the shop can observe the user via δ in the blacklist and report the transaction to the police.

VI. Comparisons

TABLE I
Feature comparisons

	AC		nTS	RWT	TP
	Revokability	Traceability			
ours	Yes	Yes	Yes	Yes	Yes
[12]	Yes	No	Yes	Yes	No
[13]	Yes	Yes	No	No	No
[14]	Yes	Yes	Yes	No	No

AC: Anonymity Control
nTS: non-TTP-Storing, **RWT**: Revokability without the help of TTP
TP: Theoretical Proof on unlinkability and unforgeability

In this section, we compare our scheme with [12] [13] [14] in some features and the computation cost. The features are described as follows.

- **Anonymity Control:**
  This contains **Revokability** and **Traceability** which have been described in Section III-B.
- **non-TTP-storing:**
  TTP does not need to store any information about the e-cash for anonymity control.
- **Revokability without the help of TTP when double-spending:**
  When a user doubly spent an e-cash, the bank can directly find the identity of the user by itself without TTP's help.

Our scheme satisfies all of above features, but the others do not. We show the comparison result in Table I.

In Table II, we measure the computation cost of the revoking procedure when double-spending, the payment protocol, and the withdrawal protocol. Besides, we also show the computation reduction percentage which is defined as $(1-\frac{A}{C})\times 100\%$ where A is the computation cost of our scheme and C is the computation cost of another scheme in Table II.

In our scheme, the bank only requires one modular inverse multiplication to revoke the anonymity of a user without the help of TTP when the user doubly spent her/his e-cash.

VII. Provable Security

In our proposed e-cash ($\mathcal{ECREC}$) scheme, there are some security issues must be considered below.

1) Unlinkability: None can trace a user's consuming behavior.
2) Unforgeability: None can issue e-cash(s) except the bank.
3) Tamper Resistance: Any information in an e-cash cannot be tampered.
4) No Swindling: Only the real owner of the e-cash can spend it.

In the followings, we give theoretical security proofs for unlinkability and unforgeability.

A. Unlinkability

In this section, we define a linkability game and show that our scheme satisfies the property of unlinkability. The linkability game is shown below.

Definition 2: *The Linkability Game*. Let k be a security parameter. Let U_0 and U_1 be two honest users and $\mathcal{J}$ be the judge that follows the $\mathcal{EDREC}$ scheme, and let $\mathcal{B}$ be the bank that is involved in the following game with U_0, U_1, and $\mathcal{J}$. The game environment is shown below.

- **Step 1:** According to the $\mathcal{EDREC}$ scheme, $\mathcal{B}$ generates the bank's public-private key $((e_b, n_b), (d_b, p_b, q_b))$, system parameters (p, q, g), and a hash function H. $\mathcal{J}$ generates the judge's public-private key $(pk\_j, sk\_j))$.
- **Step 2:** $\mathcal{B}$ generates and outputs two message pairs (r_0, m_0, y_0) and (r_1, m_1, y_1).
- **Step 3:** We randomly choose a bit $\hat{b} \in \{0,1\}$ and place $(r_{\hat{b}}, m_{\hat{b}}, y_{\hat{b}})$ and $(r_{1-\hat{b}}, m_{1-\hat{b}}, y_{1-\hat{b}})$ on

TABLE II
The Computation Cost

	Revokability		Payment				Withdrawal	
	B	R	U	R	S	R	U	R
ours	1E	0%	<1E	0%	2E	0%	4E	0%
[12]	2E	≈ 50%	<1E	≈ 0%	8E	≈ 75%	17E	≈ 76%
[13]	3E	≈ 67%	22E	> 99%	19E	≈ 90%	20E	≈ 80%
[14]	4E	≈ 75%	8E	> 88%	11E	> 82%	19E	≈ 80%

E: Modular Exponentiation
U: User, **B**: Bank, **S**: Shop
R: Computation reduction percentage: $(1-\frac{A}{C})\times 100\%$ where A is the cost of our scheme and C is the cost of another scheme

the private input tapes of U_0 and U_1, respectively. The bit $\hat{b}$ will not be disclosed to $\mathcal{B}$.

- **Step 4:** $\mathcal{B}$ performs the withdrawal protocol of the $\mathcal{EDREC}$ scheme with U_0 and U_1, respectively.
- **Step 5:** If U_0 and U_1 output two e-cash(s) which are $(\Sigma_{\hat{b}}, y_{\hat{b}}, m_{\hat{b}}, r_{\hat{b}}, \delta_{\hat{b}})$ and $(\Sigma_{1-\hat{b}}, y_{1-\hat{b}}, m_{1-\hat{b}}, r_{1-\hat{b}}, \delta_{1-\hat{b}})$ on their private tapes, respectively, we give the two 5-tuples in a random order to $\mathcal{B}$; Otherwise, $\perp$ is given to $\mathcal{B}$.
- **Step 6:** $\mathcal{B}$ outputs $\hat{b}' \in \{0,1\}$ as the guess of $\hat{b}$. $\mathcal{B}$ wins the game if $\hat{b} = \hat{b}'$. We define the advantage of $\mathcal{B}$ as

$$Adv_{\mathcal{B}}^{\textit{Linkability}}(k) = |2P[\hat{b}' = \hat{b}] - 1| \quad (3)$$

where $P[\hat{b}' = \hat{b}]$ denotes the probability of $\hat{b}' = \hat{b}$.

Definition 3: *Unlinkability*. The $\mathcal{EDREC}$ scheme satisfies the unlinkability property if the advantage $Adv_{\mathcal{B}}^{\textit{Linkability}}(k)$ is negligible.

Theorem 1: The proposed $\mathcal{EDREC}$ scheme satisfies the unlinkability property.

Proof of Theorem 1. In Step 5 of Definition 2, if $\mathcal{B}$ is given $\perp$, it will determine $\hat{b}$ with probability $\frac{1}{2}$ which is exactly the same as a random guess of $\hat{b}$.

Here, we assume that $\mathcal{B}$ gets $(\Sigma_0, y_0, m_0, r_0, \delta_0)$ and $(\Sigma_1, y_1, m_1, r_1, \delta_1)$. Let $((\beta_i, t_i), E_{pk\_j}(k_i, m_i, r_i), E_{k_i}(x_i, c_i, k_i, \delta_i))$ be the view of the data exchanged between U_i and B during the withdrawal protocol where $i \in \{0,1\}$.

Given $(\Sigma, y, m, r, \delta) \in \{(\Sigma_0, y_0, m_0, r_0, \delta_0), (\Sigma_1, y_1, m_1, r_1, \delta_1)\}$, for (β_i, t_i), $E_{pk\_j}(k_i, m_i, r_i)$, and $E_{k_i}(x_i, c_i, k_i, \delta_i)$, $i \in \{0,1\}$, there always exists a value c_i where $c_i = (\frac{\beta_i}{h_{HK}(m,r)H(\delta||y)})^{d_b} \bmod n_b$ and via $t_i = \beta_i^{d_b} \bmod n_b$, $\Sigma \equiv (h_{HK}(m,r)H(\delta||y))^{d_b} \equiv (c_i)^{-1} t_i \pmod{n_b}$ is always true. Besides, E_{pk_j} and E_{k_i} are two semantically secure encryption functions. Thus $\mathcal{B}$ cannot learn any information from $E_{pk\_j}(k_i, m_i, r_i)$ and $E_{k_i}(x_i, c_i, k_i, \delta_i)$.

Thus, the bank $\mathcal{B}$ succeeds in determining $\hat{b}$ with probability $\frac{1}{2}$. We have that $P[\hat{b}' = \hat{b}] = \frac{1}{2}$ and $Adv_{\mathcal{B}}^{\textit{Linkability}}(k) = 0$. Therefore, the proposed $\mathcal{EDREC}$ scheme satisfies the unlinkability property.

B. *Unforgeability*

M. Bellare et al. introduced a problem which is called the RSA Chosen-Target Inversion (RSA-CTI) Problem [15] in 2003 and proved that it is hard. They also provided the following theorem.

Theorem 2: Let k be a security parameter. If the RSA-CTI problem is hard, Chaum's blind signature scheme is polynomially secure against RSA one-more forgery (RSA-OMF). Concretely, for a forger $\mathcal{F}$, there exists a challenger $\mathcal{C}$ such that

$$Adv_{\mathcal{F}}^{\textit{RSA-OMF}}(k) \leq Adv_{\mathcal{C}}^{\textit{RSA-CTI}}(k) \quad (4)$$

and the time-complexity of $\mathcal{C}$ is polynomial in the time-complexity of $\mathcal{F}$ where $Adv_{\mathcal{F}}^{\textit{RSA-OMF}}(k)$ is the probability of $\mathcal{F}$ succeeding in RSA-OMF and $Adv_{\mathcal{C}}^{\textit{RSA-CTI}}(k)$ is the probability of $\mathcal{C}$ solving the RSA-CTI problem.

By Theorem 2, Chaum's RSA-based blind signature scheme [1] is secure against one-more forgery as long as the RSA-CTI problem is hard. Here, we provide a theorem to demonstrate that the proposed $\mathcal{EDREC}$ scheme satisfies unforgeability if Chaum's RSA-based blind signature scheme is secure.

Theorem 3: For any attacker $\mathcal{A}$ forging an e-cash in the proposed $\mathcal{EDREC}$ scheme, there exists a forger $\mathcal{F}$ attacking Chaum's blind signature scheme such that

$$Adv_{\mathcal{A}}^{\mathcal{EDREC}}(k) \leq Adv_{\mathcal{F}}^{\textit{RSA-OMF}}(k) \quad (5)$$

and the time-complexity of $\mathcal{F}$ is polynomial in the time-complexity of $\mathcal{A}$ where $Adv_{\mathcal{A}}^{\mathcal{EDREC}}(k)$ is the probability of $\mathcal{A}$ forging an e-cash in the proposed scheme.

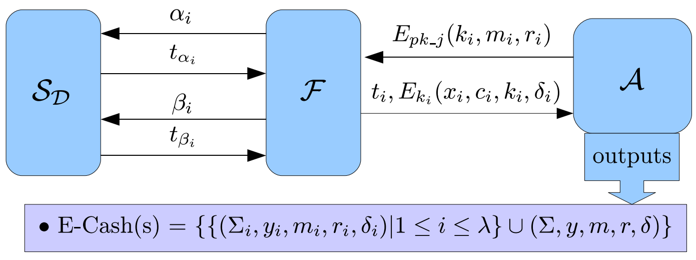
$\mathcal{S}_\mathcal{D}$
α_i
t_{α_i}
β_i
t_{β_i}
$\mathcal{F}$
$E_{pk\_j}(k_i, m_i, r_i)$
$t_i, E_{k_i}(x_i, c_i, k_i, \delta_i)$
$\mathcal{A}$
outputs
• E-Cash(s) = $\{\{(\Sigma_i, y_i, m_i, r_i, \delta_i)|1 \leq i \leq \lambda\} \cup (\Sigma, y, m, r, \delta)\}$

Fig. 4. The proof model of the unforgeability

Proof of Theorem 3. There exists an attacker $\mathcal{A}$, a forger $\mathcal{F}$, and a signing oracle $\mathcal{S}_\mathcal{D}$ of Chaum's blind signature scheme in the following game. According to the $\mathcal{EDREC}$ scheme, we generate the judge's public-private key $(pk\_j, sk\_j))$ and select a hash function H. We also randomly select two primes (p, q) such that $q|(p-1)$ and choose a generator g with order q in $\mathbb{Z}_p^*$. Let (n_b, e_b) be the public key of $\mathcal{S_D}$. We publish $(n_b, e_b, n_j, e_j, H, p, q, g)$. The model of the proof is shown in Fig. 4. $\mathcal{A}$ can query an e-cash by sending $E_{pk\_j}(k_i, m_i, r_i)$ to $\mathcal{F}$ and $\mathcal{F}$ will return $(t_i, E_{k_i}(x_i, c_i, k_i, \delta_i))$ to $\mathcal{A}$ as our proposed scheme. $\mathcal{F}$ is defined in Fig. 5. After querying $\mathcal{F}$ λ times, if $\mathcal{A}$ successfully outputs λ e-cash(s), $(\Sigma_i, y_i, m_i, r_i, \delta_i)'s$, and a forged one, $(\Sigma, y, m, r, \delta)$, where $y_i \neq y$, $m_i \neq m$, $r_i \neq r$, and $\delta_i \neq \delta$ for $i = 1, ..., \lambda$.

Thus, $\mathcal{F}$ can successfully perform one-more-forgery in Chaum's blind signature scheme via the following procedure: After receiving $(\Sigma_i,\ y_i,\ m_i,\ r_i,\ \delta_i)'s$ and $(\Sigma, y, m, r, \delta)$, $\mathcal{F}$ randomly selects $\tilde{b} \in \mathbb{Z}_{n_b}^*$ and computes $\tilde{\beta} = \tilde{b}^{e_b} H(\delta||y) \bmod n_b$. Then it sends $\tilde{\beta}$ to S_D and gets $t_{\tilde{\beta}} = \tilde{\beta}^{d_b} \bmod n_b$. It computes $\tilde{S} = \tilde{b}^{-1} t_{\tilde{\beta}} \bmod n_b$ and $S' = \Sigma \cdot (\tilde{S})^{-1} \bmod n_b$. Thus, $\mathcal{F}$ can output $(2\lambda + 2)$ Chaum's signatures $\{(s_{i_1}, (y_i, m_i, r_i))|1 \leq i \leq \lambda\} \cup \{(s_{i_2}, \delta_i)|1 \leq i \leq \lambda\} \cup \{(\tilde{S}, \delta), (S', (y, m, r))\}$ where $s_{i_1}^{e_b} \equiv h_{HK}(m_i, r_i) \pmod{n_b}$, $s_{i_2}^{e_b} \equiv H(\delta_i||y_i) \pmod{n_b}$, $\tilde{S}^{e_b} \equiv H(\delta||y) \pmod{n_b}$, and $(S')^{e_b} \equiv h_{HK}(m, r) \pmod{n_b}$. Consequently, $\mathcal{A}$ queries $\mathcal{F}$ λ times and outputs $(\lambda + 1)$ e-cash(s), and we have that $\mathcal{F}$ queries S_D $(2\lambda + 1)$ times and outputs $(2\lambda + 2)$ Chaum's signatures. Therefore, $\mathcal{F}$ succeeds in one-more forgery in Chaum's blind signature scheme, and thus $Adv_{\mathcal{A}}^{\mathcal{EDREC}}(k) \leq Adv_{\mathcal{F}}^{\textit{RSA-OMF}}(k)$.

VIII. Conclusions

We have proposed a novel off-line e-cash scheme which supports the bank to efficiently retrieve the owner's identity of an e-cash which was doubly spent. The bank can deal with the revokability issue on double-spending by itself without the participation of TTP. We also provided the formal proofs on unlinkability and unforgeability. Furthermore, we have shown the comparisons on some key features and the computation cost in Table I and Table II. Due to low computation cost, one modular inverse multipilication only, in the payment protocol, we believe that our scheme is suitable for mobile devices.

```
$\mathcal{F}(E_{pk\_j}(k_i, m_i, r_i))$;
 1. Get $E_{pk\_j}(k_i, m_i, r_i)$;
 2. Decrypt $E_{pk\_j}(k_i, m_i, r_i)$ to get $(k_i, m_i, r_i)$;
 3. Select $r_{1i}, r_{2i} \in_R \{0,1\}^{l_r}$ and $a_i, b_i \in_R \mathbb{Z}_{n_b}^*$;
 4. Compute $x_i = (ID_i \| r_{1i}) \in \mathbb{Z}_q^*$;
 5. Compute $y_i = g^{x_i} \mod p$;
 6. Compute $\delta_i = E_{pk\_j}(ID_i \| r_{2i})$;
 7. Compute $\alpha_i = a_i^{e_b} h_{HK}(m_i, r_i) \mod n_b$;
 8. Send $\alpha_i$ to $S_D$ and get $t_{\alpha_i} \equiv \alpha_i^{d_b} \pmod{n_b}$;
 9. Compute $\beta_i = b_i^{e_b} H(\delta_i || y_i) \mod n_b$;
10. Send $\beta_i$ to $S_D$ and get $t_{\beta_i} \equiv \beta_i^{d_b} \pmod{n_b}$;
11. $s_{i_1} = a_i^{-1} t_{\alpha_i} \mod n_b$;
12. $s_{i_2} = b_i^{-1} t_{\beta_i} \mod n_b$;
13. Compute $t_i = t_{\alpha_i} \cdot t_{\beta_i}$;
14. Compute $c_i = a_i \cdot b_i$;
15. Return $(t_i, E_{k_i}(x_i, c_i, k_i, \delta_i))$;
```

Fig. 5. Forger $\mathcal{F}$ in the proof of Throrem 3

References

[1] D. Chaum, "Blind signatures for untraceable payments," *Advances in Cryptology - CRYPTO'82*, pp. 199–203, 1983.

[2] C.-I. Fan and V. S.-M. Huang, "Anonymous authentication protocols with credit-based chargeability and fair privacy for mobile communications," *International Workshop on Security (IWSEC), Lecture Notes in Computer Science, Springer-Verlag*, vol. 4752, 2007.

[3] C.-I. Fan, B.-W. Lin, and S.-M. Huang, "Customer efficient electronic cash protocols," *Journal of Organizational Computing and Electronic Commerce*, vol. 17, no. 3, 2007.

[4] W.-S. Juang and H.-T. Liaw, "A practical anonymous multi-authority e-cash scheme," *Applied Mathematics and Computation, New York, Elsevier Press*, vol. 147, no. 3, 2004.

[5] S. Brands, "Untraceable off-line cash in wallet with observers," *In Advances in Cryptology - CRYPTO'93, Lecture Notes in Computer Science, Springer-Verlag*, vol. 773, pp. 302–318, 1993.

[6] C.-I. Fan and Y.-K. Liang, "Anonymous fair transaction proto-

cols based on electronic cash," *International Journal of Electronic Commerce*, vol. 13, no. 1, 2008.

[7] C. Popescu, "An off-line electronic cash system with revokable anonymity," *Proceedings of the 12th IEEE Mediterranean Electrotechnical Conference*, vol. 2, 2004.

[8] S. Miyazaki and K. Sakurai, "A more efficient untraceable e-cash system with partially blind signatures based on the discrete logarithm problem," *Financial Cryptography ,Lecture Notes in Computer Science, Springer-Verlag*, vol. 1465, 1998.

[9] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, 1978.

[10] H. Krawczyk and T. Rabin, "Chameleon signatures," *In Symposium on Network and Distributed Systems Security (NDSS '00)*, pp. 143–154, 2000.

[11] S. Pearson, "Trusted computing platforms, the next security solution," *Technical Report HPL-2002-221, Hewllet-Packard Laboratories*, 2002.

[12] J. K. Liu, P. P. Tsang, and D. S. Wong, "Recoverable and untraceable e-cash," *EuroPKI 2005, Lecture Notes in Computer Science, Springer-Verlag*, vol. 3545, pp. 206–214, 2005.

[13] W. Qiu, "A fair off-line electronic payment system," *Studies in Computational Intelligence, Springer-Verlag*, pp. 177–195, 2007.

[14] X. Hou and C. H. Tan, "Fair traceable off-line electronic cash in wallets with observers," *in Advanced Communication Technology. Phoenix Park, Korea: IEEE Computer Society*, pp. 595–599, 2004.

[15] M. Bellare, C. Namprempre, D. Pointcheval, and M. Semanko, "The one-more-rsa-inversion problems and the security of chaums blind signature scheme," *Journal of Cryptology*, vol. 16, no. 3, pp. 185–215, 2003.