An e-cash Scheme with Multiple Denominations and Transferability

Jia-Ning Luo[1], Ming-Hour Yang[2]

[1] Department of Information and Telecommunications Engineering, Ming Chuan University
[2] Department of Information & Computer Engineering, Chung Yuan Christian University
deer@mail.mcu.edu.tw, mhyang@cycu.edu.tw

Abstract

E-commerce has developed rapidly in recent years and online transactions and digital services have become popular. However, existing trading systems such as ATMs, credit cards, Paypal, and prepaid systems are potentially unsecure and users' privacy is not sufficiently protected. Because user information can be easily associated with consumption history in the aforementioned trading systems, consumers' spending habits and interests are retrieved and analyzed by individuals or businesses or people whose intentions may be suspect and without the consent of the consumers themselves. Therefore, the development of e-cash is critical for future e-commerce. In an e-cash system, blind signatures ensure consumer anonymity and prevent e-cash from being linked to its user. However, the anonymous nature of e-cash makes tracing criminal behavior difficult, so e-cash systems with anonymity revocability have been proposed. To thus expand the applicability of e-cash systems, alternatives have become the goal of subsequent studies (e.g., offline transactions, transferability, and divisible e-cash).

This study proposed an e-cash system with multiple denominations that enables e-merchants to give customers change when in an offline environment. The proposed system results in convenient transactions regardless of transaction amount and reduces the amount of e-cash users need to deposit in advance.

Keywords: Anonymity, Blind signature, e-cash, Electronic payment, Transferability

1 Introduction

Electronic cash (e-cash) protects the privacy of online transactions through anonymity. Consequently, neither banks nor merchants can analyze consumer behaviors through their e-cash transactions. Such transactions provide security and anonymity guarantees to all parties involved in a transaction [1-3].

Although e-cash transactions are similarly convenient and private to cash transactions, their anonymous nature complicates e-cash management from the issuer's perspective. Chang and Lai [4] and Fan et al. [5] proposed date-attachment e-cash schemes that allowed for expiration and deposit dates, enabling banks to manage their database growth by forcing users to renew their expired e-cash at banks. Fan et al. [6] proposed a recoverable e-cash scheme with revocation features to solve problems caused by lost or damaged e-cash devices, thereby increasing the number of channels for e-cash acceptance during online transactions and extending the life cycle of e-cash devices.

Offline transactions are necessary in mobile trading environments wherein the Internet is difficult to reach, such as on buses, trains, and airplanes [7]. However, such e-cash transactions are traceable because offline e-cash schemes [6-9] cannot guarantee anonymity and are vulnerable to double spending. In addition, Internet unavailability prevents merchants from immediately connecting to banks and ensuring whether a customer's e-cash has been revoked or double-spent. When problems such as double spending occur, e-cash anonymity makes tracing a specific double-spending user difficult. To prevent loss due to malicious offline transactions such as double spending, e-cash anonymity can be implemented through a trusted third party (TTP) [10-12]. For example, Fan and Huang [11] traced fraudulent consumers by retrieving user IDs in double-spending transactions through devices in banks placed by a TTP. Eslami and Talebi [13] and Baseri et al. [14] further proposed methods to identify fraudulent e-cash users without TTP assistance using traceable verification messages generated by secret parameters during transactions. Consequently, investigators only needed to identify the verification messages generated by any previous two transactions to obtain the user ID of a fraudulent e-cash user.

In reality, it is difficult for users to predict the amount of currency required for future transactions and buy the corresponding amount of e-cash in advance. Additionally, a user's real-time balance cannot be identified in an offline environment. Therefore, Sarkar proposed a transferable e-cash scheme [15] that

*Corresponding Author: Ming-Hour Yang; E-mail: mhyang@cycu.edu.tw
DOI: 10.3966/160792642019122007025

extends online transferable e-cash into offline environments, transferring the balance in offline transactions to consumers. However, to ensure future traceability of offline double-spending e-cash users, Sarkar's method requires the encryption of the transferring party's ID using its own public key that must be attached to the transaction record, resulting in the continuous growth of the e-cash database. In addition, because transferees cannot immediately confirm the transferor's data accuracy when offline, forged transfer data will make any double spending prior to the current transaction untraceable. Fuchsbauer et al. [16] recorded the e-cash transfer process using a group signature to solve merchants' verification problems. The transferor joined a group to obtain its private key and signed receipts containing the recipient ID. These group signatures enabled transferees to verify whether a receipt was group-signed while ensuring the signee's anonymity. When double spending occurred, the transferee identified the user IDs of all suspicious receipts through a group server and searched for the user ID of the previous user throughout the receipts stored in the database. Then, the suspicious transaction's e-cash flow was reconstructed to identify the consumer responsible for the double spending. Although enabling e-cash users to personally store partial receipts alleviated the storage problem caused by multiple transfer records, the lost user records led to the untraceability of malicious consumer behaviors.

To reduce the number of e-cash deposits and simplify transaction processes involving various transaction amounts, Okamoto [17] proposed an offline divisible e-cash scheme that expresses all possible combinations of e-cash withdrawals in binary trees. Variable payment amounts were achieved by arbitrarily dividing the denomination into subsets of any size. However, this method required substantial exponentiation and communication during e-cash transactions, and damaged consumer privacy because a consumer's prior transactions were linkable. Canard and Gouget [18] proposed an offline divisible e-cash scheme that did not exhibit linkability. Au et al. [19] used bounded accumulators to increase the binary-tree computation representing various e-cash combinations during e-cash withdrawal and reduce the required computational power and data transfer during transactions; however, this method resulted in counterfeit e-cash problems. Therefore, Canard and Gouget [20] further proposed a novel binary-tree approach to prevent counterfeiting while maintaining a similar computational complexity as the method of Au et al. [19]. Batten and Yi proposed a scheme providing can solve the change-giving problem [21].

None of these methods, however, are suitable for mobile devices with limited storage space because of the size of banks' complicated binary-tree verifications.

This study proposed an e-cash scheme with multiple denominations and applicable to both online and offline transactions. Compared with existing offline divisible e-cash schemes, the proposed scheme requires less data storage and has lower computation complexity during transactions. Banks generate balance of online transactions to reduce the data stored on users' e-cash devices, whereas merchants are responsible for the change given to consumers (i.e., e-cash balance) during offline transactions, which is provided through offline transfer. The e-cash transaction protocol proposed in this study satisfies the following offline e-cash transaction security requirements:

- Anonymity: The user's spending records are protected by the untraceability of e-cash usage even in the occurrence of collusion between the bank and merchant.
- Unlinkability: The proposed e-cash scheme ensures users' anonymity and transaction unlinkability by disallowing the identification of any similarity between e-cash users.
- Unforgeability: Valid e-cash cannot be created from known e-cash or from anywhere other than the bank itself.
- Double spending detection: The bank can detect double spending regardless of whether transactions occur online or offline.
- Anonymity revocability: When double spending occurs, the bank can revoke user anonymity using the double-spending detector provided by the TTP.
- Traceability: When double spending occurs, user anonymity revocation reveals the user's spending history through the device provided by the TTP.

The remainder of this paper is organized as follows: Section 2 introduces the research framework and the four e-cash transaction steps: system initialization, e-cash withdrawal, online or offline transaction, and redemption; Section 3 analyzes the security of the protocol and compares with that of other offline divisible e-cash protocols; Section 4 discusses the computational performance of the proposed e-cash scheme and compares with that obtained in related studies; Section 5 verifies the proposed protocol using Gong-Needham-Yahalom logic; and Section 6 concludes.

2 Multiple Denominations in E-cash with Transferability

In this section, an e-cash scheme with multiple denominations is proposed to reduce the amount of e-cash required in offline transactions. The parties involved in the proposed scheme comprise the e-cash-issuing bank, the merchant (who is not anonymous), and e-cash users.

Figure 1 illustrates the four steps of the proposed e-cash scheme from issuance to write-off: registration,

withdrawal, online or offline transaction, and redemption. When a consumer wishes to use the e-cash scheme, they must register with the bank and complete a user ID application using their mobile device. The user can withdraw e-cash multiples upon the completion of registration, and the bank issues e-cash through TTP verification, storing the user's balance on their mobile device. The online transaction protocol is implemented when users make a purchase at merchants with Internet access, wherein the purchase amount is immediately converted from e-cash to cash and transferred to the merchant's account. In the offline transaction protocol, the merchant redeems the purchase amount via the Internet at a later point in time.

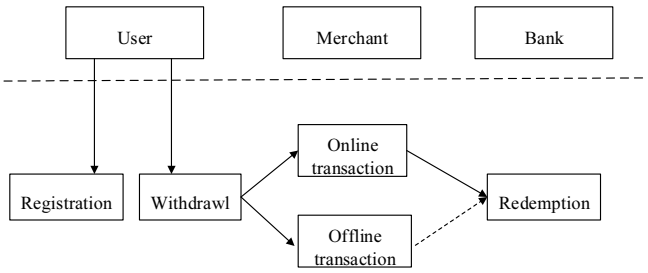

Figure 1. Flowchart of system framework

2.1 Registration

When a consumer wishes to use the e-cash scheme, they must register with the bank and complete a user ID application using their mobile device. Table 1 defines the notation used throughout this study.

Table 1. Notation used

H_1, H_2, H_3	Three hash functions
E_r, D_r	Symmetric-key function, where x is the cryptographic key
$\hat{E}_{pk}, \hat{D}_{sk}$	Cryptographic functions for public key framework, where (pk,sk) is the key pair
(pk_J, sk_J)	Public-private key pair for authentication
ID_C	User ID of User C
l_k, l_r, l_h	Secret parameters of the system, used to indicate the length of random numbers

Assume that the TTP provides the bank with a safe and reliable tamper-resistant authentication device, called as "validator". The capabilities and parameters of the validator are as follows:

- Random number generator;
- Symmetric-key cryptography;
- Asymmetric-key cryptography;
- Public-private key pair;
- Bank's public key;
- One-way hash functions H1, H2, and H3;
- Verification of business credentials.

During the initial system implementation, the bank selects large prime numbers p_b and q_b and Rivest-Shamir-Adleman (RSA) parameters $n_b = p_b q_b$ and $e_b d_b = 1(\text{mod}\ \phi(n_b))$ to generate the bank's RSA key pairs $(n_b e_b)$ and $(n_b d_b)$, , and it publishes the public keys as open information. Subsequently, the bank selects another two large prime numbers p and q, where $q \mid (p-1)$ and the element $g \in Z_p^*$ with q levels. The bank makes the following parameters public: $(n_b, e_b, p, q, g, H_1, H_2, H_3, pk_J, E, D, \hat{E}, \hat{D})$, where pk_J is the public key for authentication. Merchants register with the certificate authority in advance for authentication and select their own key pairs. The users register and select their own user ID with the bank, and save their public keys issued by the certificate authority on their mobile device for use in offline transactions.

2.2 E-cash Withdrawal

The user's credit, approved by the bank, is used to generate e-cash of corresponding value through TTP authentication and is stored on the user's mobile device, as shown in Figure 2.

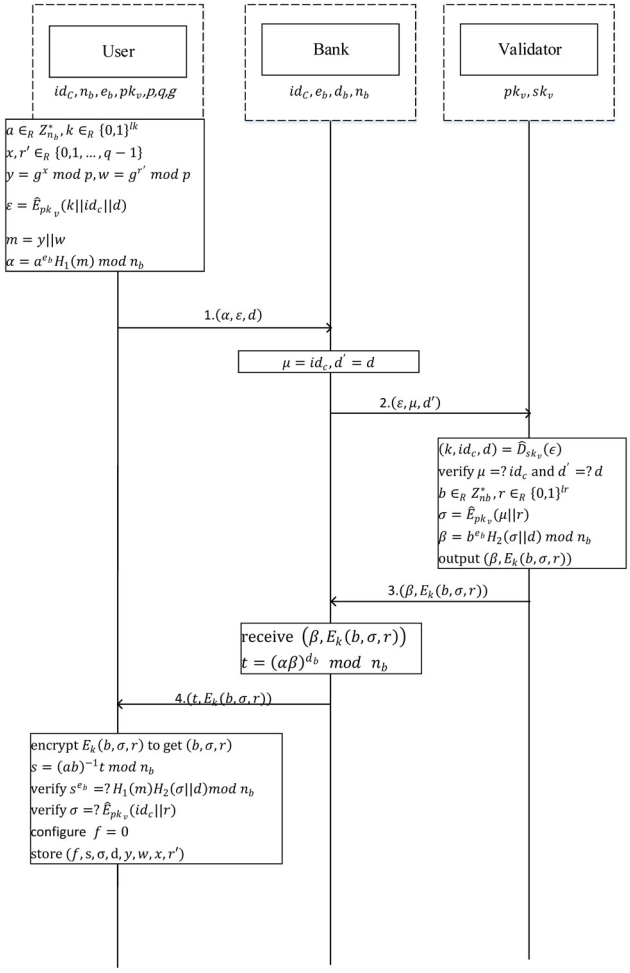

Figure 2. E-cash withdrawal procedure

User → bank. The user selects four random variables $a, k, x,$ and r', where $a \in Z_{n_b}^*, x, r' \in \{0,1,...,q-1\}$ and $k \in_R \{0,1\}^{l_k}$. Subsequently, the scheme calculates $a = a^{e_b} H_1(m) \bmod n_b$, $y = g^x \bmod p$, and $w = g^{r'} \bmod p$, where $m = y \| w; d$ is the withdrawn value; ID_c is the user ID of User C; y and w are parameters for zero-knowledge proof; k is the communication key between validator and user; and a is the ambiguous value of the information $H_1(m)$ subject to a signature with the blinding factor a added. Finally, $\in = E_{pk_j}(k \| ID_c \| d)$ is calculated before $(a, \in)$ is submitted to the bank.

Bank → validator. The bank sets the variables $d' = d$ and $\mu = ID_c$. Subsequently, the bank sends $(\in, \mu, d')$ to the validator for data authentication.

Validator → bank. The validator decrypts $\in$ using the private key and confirms whether $\mu = ID_c$ and $d' = d$ to verify that the approved value and user ID are consistent. If the authentication fails, an error message is sent to the bank to terminate the protocol; if the authentication succeeds, the validator selects a random number $b \in_R Z_{n_b}^*$ and a random text string r $\in_R \{0,1\}^{l_{r_j}}$. Subsequently, the validator calculates $\sigma = \tilde{E}_{pk_j}(\mu \| r)$ and $\beta = [b^{\in_b} H_2(\sigma \| d)^{-1}] \bmod n_b$, where b is the blinding factor for blind signature to prevent the bank's access and r is a nonce attached to the end of the user ID to ensure unlinkability by generating different values of σ. Finally, the validator calculates $\tilde{E}_k(b, \sigma, r)$ using k for symmetric-key encryption before sending both $\tilde{E}_k(b, \sigma, r)$ and β to the bank.

Bank → user. The bank issues a blind signature upon receiving $(\beta, E_k(b, \sigma, r_j))$ and combines the two data segments to calculate $t = (\alpha\beta)^{d_b} \bmod n_b$ before sending $(t, E_k(b, \sigma, r_j))$ to the user.

Unblinding. After the user receives $(t, E_k(b, \sigma, r_j))$, the communication key k can be used to decrypt (b, σ, r). The calculation of $s = (ab)^{-1} t \bmod n_b$ removes the blinding factors ab and verifies the validity of the signature $s^{\in_b} = ? H_1(m) H_2(\sigma \| d) (\bmod n_b)$. Subsequently, the user ID is verified using $\sigma = \hat{E}_{pk_j}(ID_c \| r_j)$. After the verification process is complete, a set of e-cash f = 0, (f, s, m, σ, d) is configured, wherein f is the identification tag and x and r' are stored by the user for future transactions.

2.3 E-cash Transactions

E-cash transactions comprise online and offline transactions, discussed separately in the following two subsections.

2.3.1 Offline Transactions

Figure 3 and Figure 4 illustrate how an offline transaction proceeds in the proposed e-cash scheme. The user first places an order with a merchant. After receiving the order, the merchant sends the user an invoice containing product information, consumption amount, and consumption date and including a signature using the merchant's private key. The user confirms the invoice and transfers the corresponding e-cash amount to the merchant. The merchant calculates the change owed to the user and pays the change with e-cash withdrawn in advance or e-cash obtained from prior transactions. The merchant encrypts the amount due using an attached random number σ (which identifies the e-cash user) and the public key issued by the validator before signing the proof of e-cash transfer using the merchant's private key. The e-cash change and first part of the challenge message r_c' are simultaneously transferred to the consumer. The user verifies the legitimacy of the e-cash after receiving the change, generates the final part of the challenge message r_u, and calculates the response c before sending both r_u and c back to the merchant. The merchant sends the certificate of transfer s' to the user after verifying the validity of the challenge message.

During the e-cash transfer, the merchant directly signs the e-cash with the certificate of transfer and achieves an offline transaction. The transferred e-cash must be used in online transactions to verify the merchant's signature and the correct e-cash holder. During double spending investigations, the parameters in the first e-cash transaction are subject to zero-knowledge proof, and the σ containing user ID information is added to a new set of random numbers before being attached to the certificate of transfer. Because the merchant uses its own signature as the proof of transfer, all consumption and transfer history is traceable.

User → merchant. The user chooses an item and sends an order to the merchant.

Merchant → user. After the merchant receives the order from the user, $OI = (OA, date, OD)$ is generated, where OA is the purchase amount, *date* is the time and date the order was made, and OD is the product name and product description. Subsequently, the merchant signs with its own private key to generate $Sig_s(OI)$ and sends the signature back to the user.

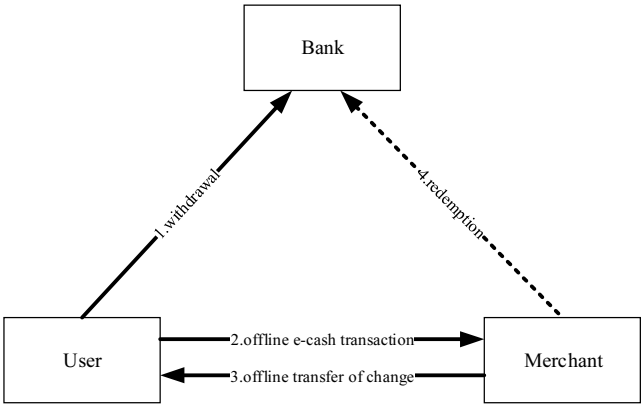

Figure 3. System framework of e-cash issuance and offline transactions

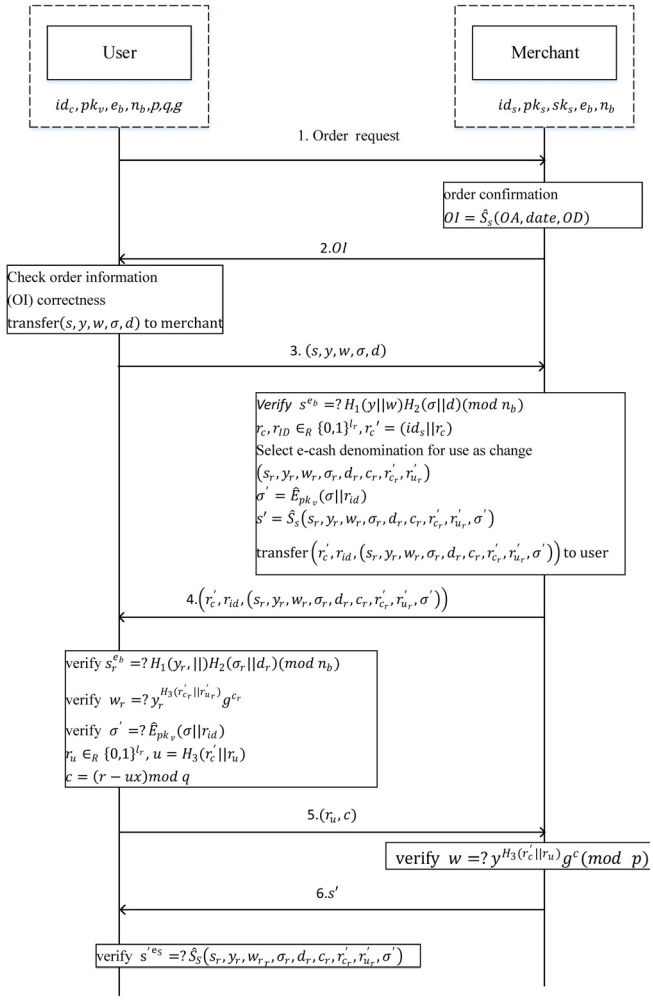

Figure 4. Flowchart of offline e-cash payment

User → merchant. Upon receiving *OI*, the user confirms the correctness of the *OD* content and *date* before selecting the e-cash balance due (s, y, w, σ, d) and sending it to the merchant.

Merchant → user. After verifying the legitimacy of the e-cash, the merchant selects two random numbers r_c and r_d, where $r_c, r_{ID} \in_R \{0,1\}^{l_r}$.. Subsequently, the merchant calculates $r_c' = (ID_s \,||\, r_c)$ as the first part of the challenge message and retrieves σ from the received e-cash for authentication. The merchant signs both the received e-cash and the calculated $\sigma = E_{pk_j}(\sigma \,||\, r_{ID})$ attached to the certificate of transfer. Finally, the merchant calculates the change owed to the user $s' = Sig_s(s_r, y_r, w_r, \sigma_r, d_r, c_r, r_{c_r}', r_{u_r}', \sigma')$ and sends the e-cash change along with r_c' and r_{ID} to the user.

User → merchant. Upon receiving the e-cash change, r_c' and r_{ID}, the user verifies the change amount and its legitimacy and the correctness of the ID containing σ'. The user then selects $r_u \in \{0,1\}^{l_r}$ and calculates $u = H_3(r_u)$ to complete the challenge message. The user employs their secret x to calculate the collision $c = (r - ux) \bmod q$ before sending (r_u, c) back to the merchant for authentication.

Merchant → user. Upon receiving the collision c, the merchant calculates whether $w = y^{H_3(r_c'||r_u)} g^c \ (mod\ q)$. If so, the zero-knowledge proof is authenticated. The merchant then sends the certificate of transfer s' to the user and stores the received e-cash for future customer change or redeems it with the bank once they have Internet access.

User. The offline transaction is complete once the user confirms the correctness of the certificate of transfer and stores it with their e-cash.

2.3.2 Online Transactions

Figure 5 and Figure 6 illustrate the procedure for online transactions in the proposed e-cash scheme. The user first places an order with the merchant. After receiving the order, the merchant sends the user an invoice containing the product information, consumption amount, and consumption date and including a signature using the merchant's private key. The user confirms the invoice and selects an amount of e-cash that corresponds to the amount due, after which a message containing the user's e-cash status is sent to the merchant. The merchant sends the message received from the user and the OI to the bank. The bank compares the OI from both the user and merchant and authenticates the user's e-cash status. If the authentication is passed, the bank checks the database to confirm whether double spending is occurring. The bank then generates new e-cash for customer change and sends it with a blind signature and message of successful transaction back to the merchant before adding the transaction amount to the merchant's account. Upon receiving the change from the bank, the merchant transfers the e-cash change to the user. The online transaction is complete once the user unblinds the change and verifies the amount of e-cash change received.

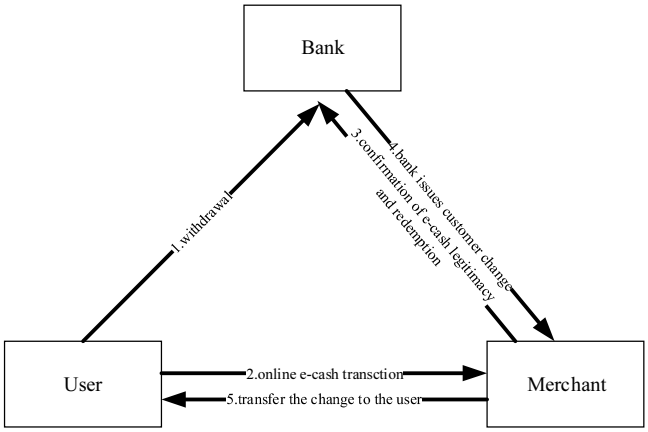

Figure 5. System framework of e-cash issuance and online transactions

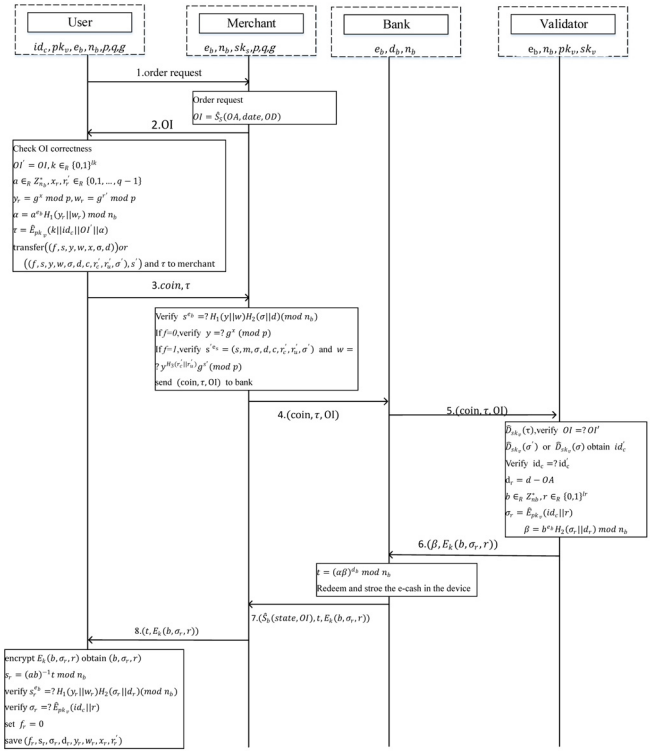

Figure 6. Message flows of online e-cash payment

User → merchant. The user chooses an item and sends an order to the merchant.

Merchant → user. Upon receiving the OI from the user, the merchant calculates $Sig_s(OI)$, where OI is the same as that defined for offline transactions. The merchant then sends OI back to the user.

User → merchant. Upon receiving the OI from the merchant, the user sets the variable $OI' = OI$ and selects e-cash for this transaction before assigning four random variables a, k, x and r', where $a \in Z_{n_b}^*$, $x, r' \in \{0,1,...,q-1\}$, and $k \in_R \{0,1\}^{l_k}$. The user then calculates $\alpha = \alpha^{\in_b} H_1(m) \bmod n_b$, $y = g^x \bmod p$ and $w = g^{r'} \bmod p$, where $m = y \| w$. Finally, the user sets $\tau = E_{pk_j}(k \| ID_c \| OI' \| \alpha)$ before sending both the e-cash and τ to the merchant.

Merchant → bank. Upon receiving the e-cash payment from the user, the merchant verifies whether $s^{\in_b} = ?H_1(m)H_2(\sigma \| d)(\bmod n_b)$. If so, the merchant performs the following verification according to f: If $f = 0$, verify $y = ?g^x (\bmod p)$; if $f = 1$, verify $s'^{\in_s} = (s, m, \sigma, d, r', r'_u, \sigma')$ and $w = ?y^{H_3(r'_c \| r'_u)} g^{s'} (\bmod p)$. If all the aforementioned verifications are passed, the merchant sends both the received e-cash and OI to the bank.

Bank → authentication device. Upon receiving the e-cash from the merchant, the bank uses its database to verify whether double spending is occurring. If double spending is not occurring, the bank sends $(coin, OI)$ to the authentication device.

Authentication device → bank. The authentication device decrypts τ and verifies whether $OI' = OI$, and then decrypts either σ or σ to obtain ID'_c according to f. Subsequently, the user verifies whether $ID_c = ?ID'_c$ to determine the legitimacy of the e-cash holder (i.e., whether the transferor and holder are the same user). Next, the authentication device calculates $d_r = d - OA$ to obtain the amount of customer change. Finally, the device generates the corresponding e-cash similarly to how it generates e-cash for user withdrawal and sends the generated $(\beta, \tilde{E}_k(b, \sigma_r, r))$ to the bank.

Bank → merchant. Upon receiving $(\beta, \tilde{E}_k(b, \sigma_r, r))$, the bank creates a blind signature for the e-cash change similarly to that created in the e-cash withdrawal process. The bank then deposits the corresponding transaction amount in the merchant's account and archives the transaction in the database before updating the transaction state and sending $(state, t, \tilde{E}_k(b, \sigma_r, r))$ to the merchant.

Merchant → user. The merchant verifies the transaction state sent by the bank. If the transaction is successful, the merchant sends the e-cash change to the user.

User → merchant. Upon receiving the e-cash change from the merchant, the user unblinds and verifies the change similarly to how withdrawals are verified, with the e-cash change stored for future use.

2.4 E-cash Redemption

The merchant sends the e-cash and all attached parameters to the bank for redemption. Upon receiving the e-cash from the merchant, the bank verifies the legitimacy of the e-cash and whether double spending has occurred using its database before signing and verifying the zero-knowledge proof. If all verifications pass, the bank deposits the corresponding e-cash value in the merchant's account and archives it in the database.

Merchant→bank:

The merchant sends its e-cash proceeds from offline transactions to the bank for redemption. The bank verifies the e-cash signature $s^{\in_b} = ?H_1(y \| w)$ $H_2(\sigma \| d)(\mathrm{mod}\, n_b)$ and the zero-knowledge proof $w = ? y^{H_3(r_c' \| r_u')} g^{s'} (\mathrm{mod}\, p)$ using its database. If no duplicate data exists, the bank deposits the corresponding cash into the merchant's account and archives the transaction for auditing purposes.

2.5 Double Spending Confirmation and Anonymous Control

Figure 7 illustrates the life cycle of e-cash in which double spending occurs.

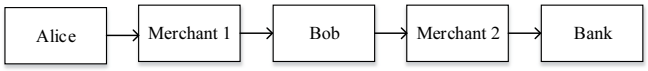

Figure 7. E-cash life cycle

User 1's double spending. When User 1 makes an e-cash payment, the zero-knowledge proofs corresponding to merchant-generated challenges are required to verify the legitimacy of the e-cash ownership. Because each challenge value is generated by the merchant and the user collectively, the user is unable to duplicate the challenge value. Therefore, if User 1 makes purchases at two merchants, the bank receives two values $(s, y, w, \sigma, d, c, r_c, r_u)$ and $(s, y, w, \sigma, d, c', r_c', r_u')$. Subsequently, User 1's identity can be decrypted using the following equation:

$$\begin{cases} c = r - ux (\mathrm{mod}\, q) \\ c' = r' - u'x' (\mathrm{mod}\, q) \end{cases}$$

where $u = H_3(r_c \| r_u)$ and $u' = H_3(r_c' \| r_u')$.

The obtained (x, r') can be sent to the authentication device along with (s, y, w, σ) to reveal User 1's identity.

Merchant 1's double spending. Merchant 1 is required to sign the certificate of transfer every time it gives a user e-cash change. Therefore, the bank can use Merchant 1's signature to verify whether Merchant 1 is responsible for double spending when the same e-cash change is given to different users.

User 2's double spending. Because User 2 is required to make an e-cash payment online, the bank immediately detects double spending when it occurs. If Merchant 1 is the victim of double spending, it can appeal to the bank for an investigation.

Merchant 2 double spending. Because Merchant 2 is required to redeem its e-cash proceeds with the bank through the Internet, the bank immediately detects double spending if Merchant 2 duplicates its redemption.

3 Security Analysis

3.1 Anonymity

When e-cash is generated, the user's ID is attached to random numbers and is encrypted to $\sigma = \hat{E}_{pk_j}(\mu \| r)$ using an validator. The user does not have access to the original content of the blind signature $t = (\alpha\beta)^{d_b} \bmod n_b$ attached by the bank. Therefore, the e-cash unblinded by the user cannot be associated with the identity of whoever withdraws the e-cash. In addition, because merchants only challenge the user through the zero-knowledge proof when the user makes a purchase, only those who have access to secret value x are capable of calculating the corresponding challenge response. Therefore, the bank uses the validator only to decrypt the user ID of suspected double spenders through the (x, r') calculated by the challenge response of duplicate spending records.

3.2 Unlinkability

Because of the anonymous nature of e-cash, any two transactions are unlinkable except when anonymity is revoked by the validator.

3.3 Unforgeability

The bank's blind signature message $t = (\alpha\beta)^{d_b} \bmod n_b$ is converted into the signature $s = (ab)^{-1} \bmod n_b$ after the user unblinds it. Ballare et al. confirmed that this RSA-based blind signature mechanism exhibits unforgeability. The bank's RSA-based blind signature cannot be forged. In addition, any modification of the parameters of issued e-cash is immediately identified when a signature is made for authentication.

3.4 Double Spending Detection

The bank is capable of detecting double spending in both online and offline transactions. Online double spending is immediately detected, whereas its own database can be used to detect offline double spending. In addition, the bank can use double spending transaction data to calculate a user's secret (x, r') and revoke the user's anonymity through the validator for further investigation.

3.5 Anonymity Revocability

When double spending occurs, the bank uses the double spending transaction data to calculate the user's secret (x, r') and revoke the user's anonymity through the validator. The bank is entitled to determine the user's secret (x, r') only if double spending occurs.

3.6 Traceability

Because the e-cash scheme proposed in this study allows only one offline transfer at a time, the merchant's signature as the proof of transfer exhibits adequate traceability while preventing storage problems caused by multiple transfers. When double spending occurs, the validator can reveal both the user's and merchant's identities.

4 Performance Comparison

This section analyzes the data storage of the proposed e-cash scheme and the computational complexity required of the bank and user during various stages of the e-cash life cycle. RSA 1024-bit keys were selected for the encryption algorithm. Multiple denominations decreased the data storage required when a large amount was paid using a single denomination. However, because the data storage required in the proposed scheme varies depending on the e-cash denomination, this study compared the scheme's e-cash data storage requirements with those of schemes proposed in previous studies. Figure 8 illustrates the data storage required by the e-cash schemes in Conrad and Gouget [20], Fan and Huang [11] and this study, wherein the denomination and data storage required (in bits) are the x- and y-axes, respectively.

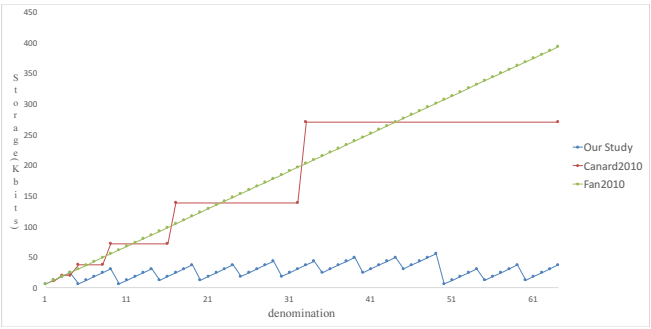

Figure 8. Comparison of e-cash data storage

In the proposed e-cash scheme, the merchant withdraws e-cash in minimum denominations in advance for offline customer change. The divisible e-cash scheme proposed by Canard and Gouget [20] has massive binary nodes that require substantial data storage, and because e-cash is withdrawn in a modular exponentiation, the line graph in Figure 8 has a stepped shape. The data storage required by the single-denomination e-cash scheme proposed by Fan and Huang [11] was linearly related to the purchase amount. By contrast, the data storage required by the proposed e-cash scheme exhibits a zigzag pattern as the e-cash amount is increased. The proposed scheme includes a variety of predefined denominations, resulting in only a slight increase in stored data as the denomination is increased. For online transactions, the bank directly generates customer's e-cash change. The denomination of customers' change is thus singular for online transactions and the required storage on users' mobile devices is substantially lower than in the other schemes.

Table 2. Comparison of e-cash computational complexity

Scheme	Fan et al.	Canard and Gouget	This study
Withdrawal	$12exp*x$	$(2^{n+3}+2^n+2-5)\ exp$ $+(n+2)Sign_{ess}$	$12exp$
Online transactions	$2exp*x$	$(3exp+2Sign_{ess}+$ $2paring)+1exp$	(16 or 18) $exp*x$
Offline transactions	$3exp*x$	$(3exp+2Sign_{ess}+$ $2paring)+1exp$	$12exp*y$
Redemption	$(2\sim3)exp*x$	$2^{l+1}exp$	$(2\sim4)\ exp*y$

Table 2 presents a comparison of e-cash computation complexity in the three aforementioned e-cash schemes, wherein $x=2^n$ is the e-cash denomination, y is the number of e-cash notes, and $sign_{ess}$ indicates the signature created using the extended special signature. This symbol *exp* and *pairing* represents an exponential operation and pairing operation, respectively. The computational complexity of the Canard and Gouget scheme [20] for online and offline transactions is equal. The scheme proposed in the present study has multiple denominations and offline transfer for merchants, so every transaction consists of n e-cash notes plus the additional computation of the customer's change. Therefore, the proposed scheme has greater computational complexity than the scheme of Fan et al. [11] per transaction, but less total computational complexity. In addition, the number of e-cash notes required for most online transactions is kept at one in the proposed e-cash scheme. The computational complexity of the scheme only increases during occasional offline transactions. Moreover, the computational complexity of the proposed scheme is fixed during e-cash withdrawal, similar to that identified in Fan and Huang [11] but requires less computational time than that the scheme presented in that study. Compared with the large-scale exponential computation required by the scheme of Canard and Gouget [20], the proposed scheme is substantially more efficient. Finally, the computational complexity of the scheme during e-cash redemption depends on the number of e-cash notes to be redeemed. Similar to the computational complexity during e-cash transactions, n is maintained as 1 on most occasions. Although the proposed scheme's computational complexity is slightly greater than that of Fan and Huang [11] per e-cash note, the total computation in the proposed scheme is less than that needed in Fan and Huang [11] during redemption. When large denominations are redeemed, the proposed scheme has similar data storage requirements to the scheme of Canard and Gouget [20], but is substantially more

efficient in terms of computational complexity.

5 Gong-Needham-Yahalom Logic Proof

In this section we use the Gong-Needham-Yahalom logic to proof the correctness of our works. The notations of the proof are listed in Table 3.

Table 3. Notations of the proof

V	Validator
B	Bank
M	Merchant
U	User
$\{X\}_K, \{X\}_K^{-1}$	Uses the symmetric key K to encrypt/ decrypt the message X.
$\{X\}_{+K}, \{X\}_{-K}$	Uses the asymmetric key K to encrypt/ decrypt the message X.
$H(X)$	Message X is protected by a one way hash function $H(X)$.
$P \lhd X$	P is told message X.
$P \in X$	P possesses message X.
$*X$	X is generated by others; $P \lhd *X$ means P is told for X which he did not convey previously.
$P \mid \equiv (X)$	P believes X is fresh. X has not been used at any time in the prior protocol, or sent by an attacker. For example, a andom number or a counter.
$P \mid \equiv (X)$	P believes X is recognizable.
$P \mid \equiv PSQ$	P believes S is shared by P and Q.
$P \mid \equiv P + KQ$	P believes Q owns the private key $-K$ correspondent to the public key $+K$.
$P \mid \equiv Q \mid \sim X$	P believes Q sent X.

5.1 Initial Assumption

- Validator

  $V \in pk_v$

  $V \in sk_v$

  $V \mid \equiv V \xleftrightarrow{pk_v} B$

  $V \mid \equiv V \xleftrightarrow{pk_v} M$

  $V \mid \equiv V \xleftrightarrow{pk_v} U$

- Bank

  $B \in pk_b$

  $B \in sk_b$

  $B \mid \equiv B \xleftrightarrow{pk_b} V$

  $B \mid \equiv B \xleftrightarrow{pk_b} M$

  $B \mid \equiv B \xleftrightarrow{id_c} U$

  $B \mid \equiv B \xleftrightarrow{pk_b} U$

- Merchant

  $M \in pk_m$

  $M \in sk_m$

  $M \mid \equiv B \xleftrightarrow{pk_b} V$

  $M \in \{s_r, y_r, w_r, \sigma_r, d_r, c_r, r'_{c_r}, r'_{u_r}\}$

  $M \mid \equiv M \xleftrightarrow{pk_m} V$

  $M \mid \equiv M \xleftrightarrow{pk_m} B$

- User

  $U \in id_c$

  $U \in pk_u$

  $U \in pk_b$

  $M \in \{f, s, y, w, d, c, r'_c, r'_u, \sigma', s'\}$

  $U \mid \equiv U \xleftrightarrow{id_c} B$

5.2 Goal

Withdrawal. The user believes the e-cash is issued by the bank.

$-(\text{Goal } 1.1), U \mid \equiv B \mid \sim \#\{s, \sigma\}$

$-(\text{Goal } 1.2), U \mid \equiv B \mid \sim \phi\{s, \sigma\}$

Offline transaction.

$-(\text{Goal } 2.1), U \mid \equiv M \mid \sim \#\{OI\}$

$-(\text{Goal } 2.2), M \mid \equiv \phi(s)$

$-(\text{Goal } 2.3), M \mid \equiv \#(c)$

$-(\text{Goal } 2.4), M \mid \equiv \phi(c)$

$-(\text{Goal } 2.5), U \mid \equiv M \mid \sim \#(s)$

$-(\text{Goal } 2.6), U \mid \equiv M \mid \sim \phi(s')$

The merchant believes that the e-cash sent by the user is legal. The user believes that the merchant transfers the change and transfers the certificate. Finally, they believe that all the messages are fresh and are recognizable.

Online transaction. The user believes the order from the merchant. The merchant believes that the e-cash sent by the user is legal. The validator trusts the order and the user identity. The merchant believes the results of the transactions. The user believes that the merchant transfers the change. Finally, two parties believe all the messages are fresh and are recognizable.

$-(\text{Goal } 3.1), U \mid \equiv M \mid \sim \#\{OI\}$

$-(\text{Goal } 3.2), M \mid \equiv \phi(s)$

$-(\text{Goal } 3.3), M \mid \equiv \phi(x)$

$-(\text{Goal } 3.4), M \mid \equiv \phi(s)$

$-(\text{Goal } 3.5), M \mid \equiv \phi(r'_c \| r'_u)$

$-(\text{Goal } 3.6), M \mid \equiv B \mid \#(state)$

$-(\text{Goal } 3.7), U \mid \equiv B \mid \sim \#\{s_r, \sigma_r\}$

$-(\text{Goal } 3.8), U \mid \equiv B \mid \sim \phi\{s_r, \sigma_r\}$

5.3 Proof

Withdrawl.

(Message 1.1) The bank knows the identity of the user from pre-existing authentication procedure.

$-U \mid \equiv \#(k, x, r')$

$-B \lhd *\{\alpha, \{k, id_c, d\}_{+pk_v}, d\}$

$-B \lhd \{\alpha, \{k, id_c, d\}_{+pk_v}, d\}$

$-B \mid\equiv U \mid\sim \{\alpha, \{k, id_c, d\}_{+pk_v}, d\}$

$-B \mid\equiv \#(\mu, d')$

$-B \mid\equiv \phi(\mu, d')$

(Message 1.2) The validator compares the information from the bank and the user, and the validator believes (k, id_c, d).

$-V \lhd *\{\{k, id_c, d\}_{+pk_v}, \mu, d'\}$

$-V \lhd \{\{k, id_c, d\}_{+pk_v}, \mu, d'\}$

$-V \lhd \{k, id_c, d\}$

$-V \mid\equiv \phi \{k, d, id_c\}$

$-V \mid\equiv \#(b, r)$

$-V \mid\equiv \#(\sigma)$

$-V \mid\equiv \#(\beta)$

(Message 1.3) The bank believes the messages are passed by the validator, and the bank and the validator have a secure channel.

$-B \lhd *\{\beta, \{b, \sigma, r\}_k\}$

$-B \lhd \{\beta, \{b, \sigma, r\}_k\}$

$-B \mid\equiv V \mid\sim \#\{\beta, \{b, \sigma, r\}_k\}$

$-B \mid\equiv \#(t)$

(Message 1.4) The user can recognizes (s, σ) and believes that the message is sent from the bank and is fresh.

$-U \lhd *\{t, \{b, \sigma, r\}_k\}$

$-U \lhd \{t, \{b, \sigma, r\}_k\}$

$-U \lhd (b, \sigma, r)$

$-U \mid\equiv \phi(s, \sigma r)$

$-U \mid\equiv B \mid\sim \#\{s, \sigma\}$(Goal 1.1)

$-U \mid\equiv B \mid \phi \#\{s, \sigma\}$(Goal 1.2)

Offline transaction.

(Message 2.1) The user can identify the order information OI that is signed by the merchant and believes OI is fresh.

$-U \lhd *(OI)$

$-U \lhd (OI)$

$-U \mid\equiv M \mid\sim \#\{O, I\}$(Goal 2.1)

$-U \mid\equiv M \mid\sim \phi\{O, I\}$

(Message 2.2) The merchant can verify the signature s.

$-M \lhd *\{s, y, \omega, \sigma, d\}$

$-M \lhd \{s, y, \omega, \sigma, d\}$

$-M \mid\equiv \phi(s)$(Goal 2.2)

$-M \mid\equiv \#(r_c, r_{id}, r_c')$

$-M \mid\equiv \#(\sigma')$

$-M \mid\equiv \#(s')$

(Message 2.3) The user can identify the signature s of the change, and the zero knowledge proof c_r and the token σ'.

$-U \lhd *\{r_c', r_{id}', s_r, y_r, w_r, \sigma_r, d_r, c_r, r_{c_r}', r_{u_r}', \sigma'\}$

$-U \lhd \{r_c', r_{id}', s_r, y_r, w_r, \sigma_r, d_r, c_r, r_{c_r}', r_{u_r}', \sigma'\}$

$-U \mid\equiv \phi(s_r)$

$-U \mid\equiv \phi(c_r)$

$-U \mid\equiv \phi(\sigma')$

$-U \mid\equiv \#(r_u)$

$-U \mid\equiv \#(u)$

$-U \mid\equiv \#(c)$

(Message 2.4) The merchant can recognize the zero knowledge proof message c.

$-U \lhd *\{r_c, c\}$

$-U \lhd \{r_c, c\}$

$-U \mid\equiv \#(c)$(Goal 2.3)

$-U \mid\equiv \phi(c)$(Goal 2.4)

(Message 2.5) The user believes the message s' is fresh and is recognizable.

$-U \lhd *\{s'\}$

$-U \lhd \{s'\}$

$-U \mid\equiv M \mid\sim \#\{s'\}$(Goal 2.5)

$-U \mid\equiv M \mid\sim \phi\{s'\}$(Goal 2.6)

Online transaction.

(Message 3.1) The user believes the order information OI is fresh and is recognizable.

$-U \in k, x_r, x_r'$

$-U \lhd *\{OI\}$

$-U \lhd \{OI\}$

$-U \mid\equiv M \mid\sim \#(OI)$(Goal 3.1)

$-U \mid\equiv M \mid\sim \phi(OI)$

$-U \mid\equiv \#(\alpha)$

$-U \mid\equiv \#\{\{k, id_c, OI', \alpha\}_{+pk_v}\}$

(Message 3.2.1) The merchant carries out different steps based on the messages sent from the user, and the merchant recognizes s and x.

$-M \lhd *\{f, s, y, w, x, \sigma, d\{k, id_c, OI', \alpha\}_{+pk_v}\}$

$-M \lhd \{f, s, y, w, x, \sigma, d\{k, id_c, OI', \alpha\}_{+pk_v}\}$

$-M \mid\equiv \phi(s)$(Goal 3.2)

$-M \mid\equiv \phi(x)$(Goal 3.3)

(Message 3.2.2) The merchant carries out different steps based on the messages sent from the user, and the merchant recognizes s, s' and (r_c' ||r_u').

$-M \lhd *$

$\{f, s, y, w, x, \sigma, d, c, r_c', r_u', \sigma', s', \{k, id_c, OI', \alpha\}_{+pk_v}\}$

$-M \lhd$

$\{f, s, y, w, x, \sigma, d, c, r_c', r_u', \sigma', s', \{k, id_c, OI', \alpha\}_{+pk_v}\}$

$-M \mid\equiv \phi(s')$(Goal 3.4)

$-M \mid\equiv \phi(r_c' \| r_u')$(Goal 3.5)

(Message 3.3) The symbol coin refers to the e-cash received by the merchant in the messages 3.2.1 and 3.2.2. The bank receives the e-cash and believes this is fresh.

(Message 3.4)
The validator can recognize OI, k, and id_c.

$-V \triangleleft *\{coin, \{k, id_c, OI', \alpha\}_{+pk_v}, OI\}$

$-V \triangleleft \{coin, \{k, id_c, OI', \alpha\}_{+pk_v}, OI\}$

$-V \triangleleft \{k, id_c, OI', \alpha\}$

$-V \mid \equiv \phi(OI)$

$-V \mid \equiv \phi(k, id_c)$

$-V \mid \equiv \#(d_r)$

$-V \mid \equiv \#(b, r)$

$-V \mid \equiv \#(\sigma)$

$-V \mid \equiv \#(\beta)$

(Message 3.5) The bank believes that the message passed by the validator, and the bank and the validator has a secure channel.

$-B \triangleleft *\{\beta, \{b, \sigma, r\}_k\}$

$-B \triangleleft \{\beta, \{b, \sigma, r\}_k\}$

$-B \mid \equiv \#(t)$

$-B \mid \equiv \#(state)$(Goal 3.6)

(Message 3.6) The merchant can verify the transaction state state of the bank

$-M \triangleleft *\{\hat{S}_b(state, OI), t, \{b, \sigma, r\}_k\}$

$-M \triangleleft \{\hat{S}_b(state, OI), t, \{b, \sigma, r\}_k\}$

$-M \mid \equiv \#(state)$

$-M \mid \equiv \phi(state)$

(Message 3.7) The user can identify (s_r, σ_r) and believes that this message is passed by the bank and it is fresh.

$-U \triangleleft *\{t, \{b, \sigma_r, r\}_k\}$

$-U \triangleleft \{t, \{b, \sigma_r, r\}_k\}$

$-U \triangleleft \{b, \sigma_r, r\}$

$-U \mid \equiv \phi(s_r, \sigma_r)$

$-U \mid \equiv B \mid \sim \#(s_r, \sigma_r)$(Goal 3.7)

$-U \mid \equiv B \mid \sim \phi(s_r, \sigma_r)$(Goal 3.8)

6 Conclusion

This study proposed an e-cash scheme suitable for both online and offline transactions. The proposed scheme requires less data storage and involves lower computational complexity than other divisible e-cash schemes because it enables merchants to transfer e-cash offline using signatures and gives customer e-cash change also offline due to its multiple-denomination design. The reduced data storage and computational complexity make it suitable and convenient for e-cash usage with mobile devices. The proposed scheme has various security measures: unlinkability, verifiability, unforgeability, double spending detection, tamper resistance, and nonrepudiation. When double spending occurs, the bank is entitled to trace fraudulent users and revoke their anonymity using an authentication device without compromising the anonymity of nonfraudulent users.

Acknowledgements

The authors gratefully acknowledge the support from Taiwan Information Security Center (TWISC) and Ministry of Science and Technology under the grants no. MOST 108-2218-E-011-021, MOST 108-2221-E-033-016, and MOST 107-2221-E-130-001.

References

[1] R. S. Anand, C. E. V. Madhavan, An Online, Transferable e-cash Payment System, *International Conference on Cryptology in India*, Calcutta, Inida, 2000, pp. 93-103.

[2] B. Carbunar, W. L. Shi, R. Sion, Conditional e-payments with Transferability, *Journal of Parallel and Distributed Computing*, Vol. 71, No. 1, pp. 16-26, January, 2011.

[3] S. Solat, *Security of Electronic Payment Systems: A Comprehensive Survey*, arXiv:1701.04556, 2017.

[4] C.-C. Chang Y.-P. Lai, A Flexible Date-attachment Scheme on e-cash, *Computers & Security*, Vol. 22, No. 2, pp. 160-166, February, 2003.

[5] C.-I. Fan, W.-Z. Sun, H.-T. Hau, Date Attachable Offline Electronic Cash Scheme, *The Scientific World Journal*, Vol. 2014, Article ID 216973, May, 2014.

[6] C.-I. Fan, V. S.-M. Huang, Y.-C. Yu, User Efficient Recoverable Off-line e-cash Scheme with Fast Anonymity Revoking, *Mathematical and Computer Modelling*, Vol. 58, No. 1, pp. 227-237, July, 2013.

[7] H. Tewari, A. Hughes, *Fully Anonymous Transferable Ecash*, IACR Cryptology ePrint Archive, 2016.

[8] C.-L. Chen, J.-J. Liao, A Fair Online Payment System for Digital Content via Subliminal Channel, *Electronic Commerce Research and Applications*, Vol. 10, No. 3, pp. 279-287, May-June, 2011.

[9] X. Hou, C.-H. Tan, Fair Traceable Off-line Electronic Cash in Wallets with Observers, *6th International Conference on Advanced Communication Technology*, Vol. 2, Phoenix Park, South Korea, 2004, pp. 595-599.

[10] J. Camenisch, U. Maurer, M. Stadler, Digital Payment Systems with Passive Anonymity-revoking Trustees, *Journal of Computer Security*, Vol. 5, No. 1, pp. 69-89, January, 1997.

[11] C.-I. Fan, V. S.-M. Huang, Provably Secure Integrated On/Off-line Electronic Cash for Flexible and Efficient Payment, *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, Vol. 40, No. 5, pp. 567-579, September, 2010.

[12] C. Popescu, An Off-line Electronic Cash System with Revokable Anonymity, *12th IEEE Mediterranean Electrotechnical Conference (MELECON 2004)*, Vol. 2, Dubrovnik, Croatia, 2004, pp. 763-767.

[13] Z. Eslami, M. Talebi, A New Untraceable Off-line Electronic Cash System, *Electronic Commerce Research and Applications*, Vol. 10, No. 1, pp. 59-66, January-February,

2011.

[14] Y. Baseri, B. Takhtaei, J. Mohajeri, Secure Untraceable Off-line Electronic Cash System, *Scientia Iranica*, Vol. 20, No. 3, pp. 637-646, June, 2013.

[15] P. Sarkar, Multiple-use Transferable e-cash, *International Journal of Computer Applications*, Vol. 77, No. 6, pp. 1-4, September, 2013.

[16] G. Fuchsbauer, D. Pointcheval, D. Vergnaud, Transferable Constant-size Fair e-cash, *International Conference on Cryptology and Network Security*, Kanazawa, Japan, 2009, pp. 226-247.

[17] T. Okamoto, An Efficient Divisible Electronic Cash Scheme, in *Annual International Cryptology Conference*, Santa Barbara, CA, USA, 1995, pp. 438-451.

[18] S. Canard, A. Gouget, Divisible e-cash Systems Can be Truly Anonymous, *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Barcelona, Spain, 2007, pp. 482- 497.

[19] M.-H. Au, W. Susilo, Y. Mu, Practical Anonymous Divisible e-cash from Bounded Accumulators, *International Conference on Financial Cryptography and Data Security*, Cozumel, Mexico, 2008, pp. 287-301.

[20] S. Canard, A. Gouget, Multiple Denominations in e-cash with Compact Transaction Data, *International Conference on Financial Cryptography and Data Security*, Tenerife, Canary Islands, 2010, pp. 82-97.

[21] L. Batten, X. Yi, Off-line Digital Cash Schemes Providing Untraceability, Anonymity and Change, *Electronic Commerce Research*, Vol. 19, No. 1, pp. 81-110, March, 2019.

Biographies

Jia-Ning Luo holds a Ph.D. degree in Computer Science of National Chiao Tung University, Taiwan. He specializes in network security, operating systems, network administration and network programming. He is currently an associate professor at department of information and telecommunications, Ming Chuan University, Taiwan. His interesting topics includes NFC-based protocols, IoT security and eWallet security.

Ming-Hour Yang received his doctoral degree in Computer Science & Info. Engineering at National Central University, Taiwan. His research mainly focuses on network security and system security with particular interests on security issues in RFID and NFC security communication protocols. Topics include: mutual authentication protocols; secure ownership transfer protocols; polymorphic worms; tracing mobile attackers.