A Transferable E-Cash Payment System Based on a Profitable Model

Fuw-Yi Yang, Su-Hui Chiu, and Chih-Wei Hsu

Abstract—E-Cash payment systems have been widely studied, and are commonly used for payments in e-commerce. They facilitate the completion of Internet-based transactions; however, they have yet to be deployed on a large scale. Bitcoins have recently been increasingly used in businesses transactions, and have caught public attention. Researchers believe that the properties involving verifiability, unforgeability, divisibility, fungibility, prevention of double spending, transferability, and a profitable payment model have been crucial to the success of Bitcoin. However, the Bitcoin payment system also has various drawbacks. This paper proposes a payment scheme that addresses these crucial properties, and provides more favorable features for an e-cash payment system. Finally, the proposed system is similar to a cash payment system, except that business is transacted over the Internet, and e-cash is used, instead of cash.

Index Terms—anonymity, decentralized payment processing, divisibility, double spending, e-cash, e-commerce, fungibility, transferable e-cash, unlikability, untraceability.

I. INTRODUCTION

With the rapid development of network technology, numerous Internet-based e-commerce services have become extensively used; these include e-cash payment systems, e-auction systems, and e-voting. Numerous applications currently involve e-cash payments, including Easy Card for public transport [31-32] and i-cash for purchasing commodities in convenience stores and supermarkets [1, 34]. These payment systems involve storing e-cash on a smart card. Therefore, as long as e-cash is stored on the card, transactions can be completed. These applications solve numerous inconveniences involving conventional money. That smart phones have a similar central processing unit (CPU) to smart cards inspired the concept of integrating mobile devices into e-cash payment systems. The rapid development of smart phones has thus contributed to the rapid development of e-cash payment systems. Therefore, e-cash payment systems are increasingly used.

Traditional e-cash payment systems involve three entities: the bank, consumer, and merchant. The processing of e-cash can be divided into three stages: the withdrawal stage, payment stage, and deposit stage. In 1983, Chaum [15] first proposed a blind-signature-based e-cash payment system. This system was characterized by anonymity, verifiability, and unforgeability, and these characteristics have subsequently been adopted in the security features of e-cash payment schemes [5-6, 9, 14, 16, 19-21, 24-28, 36-39, 45, 51-52, 56-57, 59, 61, 65].

Whether e-cash is transferable or non-transferable is a crucial criterion in an e-cash payment system. For example, a bank issues non-transferable e-cash to a consumer (the owner of e-cash) in the withdrawal stage. In the payment stage, the consumer pays e-cash to a merchant in return for a commodity. In the deposit stage, the merchant exchanges e-cash for cash. The life cycle of non-transferable e-cash payment systems is short; the progression covers withdrawal, payment, and deposit. To prevent the consumer from re-spending already-spent e-cash, the merchant must complete the deposit stage after receiving the e-cash. Therefore, for each instance of spent e-cash, the bank must complete the withdrawal and deposit stages. The bank’s computational burden is substantial if the payment system is used on a large scale. Therefore, the literature [7, 12-13, 17, 50, 53] has proposed transferable e-cash. “Transferable” means that, when the consumer pays e-cash to the merchant, the merchant can exchange it for cash, or keep it for later circulation, thereby reducing the bank’s computational burden.

E-Cash payment systems are classified into two types: online and off-line. In online payment systems [16, 44, 53, 57, 59, 61], a trusted third party (TTP) guarantees the freshness of the e-cash when a payer transfers it to a payee. The TTP is usually the bank that issues the e-cash. “Fresh e-cash” is e-cash that has not been spent previously. A bottleneck in online payment systems may result from determining the freshness of e-cash, because all traffic is directed to the TTP. In off-line payment systems [6, 9, 19, 24, 37, 39, 51], no third party participates in the transactions between the payer and the payee. A payee is unable to determine whether an e-cash is fresh until depositing the e-cash in a bank. If the double spending of e-cash occurs, the bank discloses the identity of the payer. However, the payee might still bear the financial loss.

In constructing a practical e-cash payment system, several basic security requirements must be considered [54], some of which are described in the following:

Fuw-Yi Yang is with the Computer Science and Information Engineering Department, Chaoyang University of Technology, Taichung, Taiwan (e-mail: yangfy@cyut.edu.tw).
Su-Hui Chiu is with the Office of Accounting, Chaoyang University of Technology, Taichung, Taiwan (e-mail: suhui@cyut.edu.tw).
Chih-Wei Hsu is with the Computer Science and Information Engineering Department, Chaoyang University of Technology, Taichung, Taiwan (e-mail: s10027608@cyut.edu.tw).
• Anonymity: A consumer receives e-cash from a bank after completing the withdrawal stage. However, inspecting an e-cash should learn no information about who withdrew the e-cash. This means that there is no relationship between e-cash and its owner.

• Verifiability: E-Cash is publicly verifiable.

• Unforgeability: E-Cash is generated only by authorized banks or parties. Unauthorized entities cannot forge e-cash.

• Untraceability: The identity of the payer remains confidential.

• Unlinkability: It is impossible to determine whether two payments originated from the same payer.

• Prevention of double spending: E-Cash is essentially digital information, and is thus reproducible. Therefore, certain mechanisms must prevent e-cash from being spent repeatedly.

Based on a profitable model, this paper proposes an online e-cash payment system with transferable e-cash. The proposed system not only satisfies the mentioned security requirements but also exhibits other novel advantages, i.e., exchange between different e-cash denominations, decentralized payment processing, a profitable payment model, and environmental friendliness.

The remainder of this paper is organized as follows: Section II introduces relevant information and techniques; Section III presents the proposed transferable e-cash payment system based on a profitable model; Sections IV and V provide an analysis of the performance of the system and security implications; and lastly, Section VI offers a conclusion.

II. DISCUSSION ON RELATED INFORMATION AND TECHNIQUES

A. E-Cash Payment System

E-Cash payment systems facilitate electronic payments based on a digital signature. They involve three major entities: the consumer, bank, and merchant. First, the consumer exchanges actual money for e-cash. The consumer subsequently uses the e-cash to conduct commercial online transactions. E-Cash is usually deposited in hardware devices, which are also used to complete payments during transactions. Payment is divided into three stages: the withdrawal stage, payment stage, and deposit stage. The complete process is described as follows:

• Withdrawal stage: The consumer registers at the bank, and saves money in his or her account. The consumer and the bank interactively execute a set of specific instructions for withdrawing e-cash. After completing the withdrawal stage, the consumer receives e-cash, and pays cash to the bank. Finally, the e-cash is stored in a hardware device, for example, a smart card, a smart phone, or another storage device.

• Payment stage: To conduct a transaction, the consumer transfers e-cash to a merchant for payment. After confirming the legality and freshness, the merchant accepts the e-cash, and completes the transaction. Otherwise, the merchant aborts the transaction.

• Deposit stage: To exchange e-cash for cash, the e-cash owner transfers e-cash to the bank. The bank confirms its

B. Bilinear Pairing

In bilinear pairing cryptosystems \([4, 8, 10]\), a mapping function \(e: G_1 \times G_1 \rightarrow G_2\) plays the main role. The function \(e\) maps the elements of a cyclic additive group \(G_1\) into another cyclic multiplicative group \(G_2\). Both groups have the same order \(q\). The value of \(q\) is a prime number that is sufficiently large to render infeasible solving the elliptic-curve discrete logarithm in \(G_1\) and \(G_2\). The mapping function \(e\) is considered bilinear pairing if it exhibits the following characteristics:

• Bilinearity: Let \(P, P_1\), and \(P_2\) be arbitrary elements of group \(G_1\). In addition, \(a\) and \(b\) are the arbitrary elements of \(Z_q\). Then \(e(P_1 + P_2, P) = e(P_1, P)e(P_2, P)\) and \(e(aP, bP) = e(P, P)^{ab}\).

• Non-degeneracy: The elements \(P\) and \(Q\) in the cyclic group \(G_1\) meet \(e(P, Q) \neq 1\), where \(1\) is the identity element of cyclic group \(G_2\).

• Calculability: The elements \(P\) and \(P_1\) are arbitrary elements of cyclic group \(G_1\), and an effective algorithm exists to calculate \(e(P, P_1)\).

C. Identity-Based Digital Signature from Pairings

A TTP selects the cyclic groups \(G_1\) and \(G_2\) as well as bilinear mapping \(e\), as stated in Subsection II.B. \(G_1\) and \(G_2\) have the same order \(q\). A random number \(x \in \mathbb{Z}_q^*\) is selected for the private key of the TTP, and the public key is \(P_\text{pub} = xP\), where \(P\in\mathbb{G}_1\). The TTP also selects cryptographic hash functions \(H_1(\cdot): \{0, 1\}^* \rightarrow G_1\) and \(H_2(\cdot): \{0, 1\}^* \rightarrow \mathbb{Z}_q\) for computing a message digest. Finally, the system parameters \{\(e, G_1, G_2, H_1, H_2, P, q\}\) are released.

Let \(ID_U\) be a string denoting the identity of a user \(U\). After registering with the TTP, user \(U\) receives a secret key \(X_U = xQ_U \in G_1\) for signature generation, where \(Q_U = H_1(ID_U) \in G_1\) is the user’s public key for signature verification.

To sign a message \(m\), signer \(U\) executes the following steps:

1. Choose a random number \(r \in \mathbb{Z}_q^*\) and compute \(R = rP \in G_1\).
2. Compute the message digest, \(h = H_2(m, R)\).
3. Compute the quantity \(S = rQ_U + hX_U\).

The pair \((R, S)\) is a signature on the message \(m\). Anyone who receives \((ID_U, m, R, S)\), should adhere to the following steps to verify the validity of the signed message \((m, R, S)\):

1. Compute \(h = H_2(m, R), Q_U = H_1(ID_U) \in G_1\).
2. Compute the quantities \(A = e(S, P) \in G_2\) and \(B = e(Q_U, R + hP_\text{pub}) \in G_2\).

If \(A = B\), this means that signer \(U\) has already generated the signed message \((m, R, S)\). Otherwise, \(U\) has not generated \((R, S)\) as a signature on \(m\).

D. Identity-Based Blind Signature from Pairings

Assume that a user asks a signer to sign message \(m\), but the signer is prohibited from acquiring any information regarding the message. Traditional signatures clearly cannot satisfy this specification. Chaum [15] first proposed a blind signature
scheme to fulfill this requirement. Blind signature schemes enable users to blind the messages being signed, and reshape the outside of signatures so that the signer cannot link the signatures to the users. It is a useful building block in applications where anonymity is pivotal, such as e-cash and voting systems. The following is an identity-based blind signature scheme.

To request a blind signature on message \( m \), the user and signer interactively execute the following steps:
1. Signer \( U \) chooses a random number \( r' \in \mathbb{Z}_q \), computes \( r' = r'P \in G_1 \), and sends \( r' \) to the user.
2. The user selects the random numbers \( a, b \in \mathbb{Z}_q \). The user computes \( R = ar' + bP \) and the hash value \( h = H_2(m, R) \). The user sends \( h' = h / a \mod q \) to signer \( U \).
3. Upon receiving \( h' \), signer \( U \) computes the quantity \( S' = r'(hinf + 1)Q_U + h'X_U \) and sends \( S' \) back to the user.
4. The user computes \( S = aS' + (hinf + 1)Q_U \). Therefore, \( (R, S) \) is signer \( U \)'s partially-blind signature on the blind message \( m \) and agreed common information \( info \).

When a receiver receives a partially-blind signed message, \( (ID_U, info, m, R, S) \), he or she verifies its validity according to the following steps:
1. Compute \( h = H_2(m, R), hinf = H_2(info) \), and \( Q_U = H_1(ID_U) \in G_1 \).
2. Compute the quantities \( A = e(S, P) \in G_2 \) and \( B = e(Q_U, (hinf + 1)R + hPpub) \in G_2 \). If \( A = B \), then the partially-blind signed message is valid. Otherwise, it is invalid.

The correctness of verification is as follows.
\[
S = aS' + (hinf + 1)Q_U = ar'(hinf + 1)Q_U + ah'X_U + b(hinf + 1)Q_U = (hinf + 1)(ar'Q_U + bQ_U) + hxQ_U
\]
\[
e(S, P) = e(ar'Q_U + hxQ_U + bQ_U, P) = e(Q_U, (hinf + 1)R + hPpub)
\]

E. Identity-Based Partially Blind Signature from Pairings

It might not be a good idea to blind everything in every application. In e-cash schemes, a database is required to store the deposited e-cash (coins) to detect double spending. In e-cash systems based on the blind signature scheme, the coins are usually the blind signatures issued by the bank. Therefore, without an explicit expiry date, the database will grow unlimitedly. In addition, the banks usually issue coins of different denominations to enable exact payments. Clearly inscribing the value of each coin is required.

The partially-blind signature scheme \([2, 3, 58]\) facilitates solving the aforementioned problems. A scheme based on the RSA assumption that was introduced in \([2]\) allows the blind signatures to explicitly contain some information that the two parties (user and signer) have agreed on. This can include the expiry date, denominational data, and other useful data. Based on the hardness of solving discrete logarithms, the scheme in \([3]\) is secure as long as the issued blind signatures involve logarithmic numbers. The following is an ID-based, partially-blind signature scheme.

Assume that user and signer \( U \) have agreed on the common information \( info \). To request a partially-blind signature on message \( m \), the user and signer interactively execute the following steps:
1. Signer \( U \) chooses a random number \( r' \in \mathbb{Z}_q \), computes \( hinf = H_2(info) \), \( R' = r'P \in G_1 \), and sends \( R' \) to the user.
2. The user selects random numbers \( a, b \in \mathbb{Z}_q \). He or she computes \( R = ar' + bP \) as well as the hash values \( h = H_2(m, R) \) and \( hinf = H_2(info) \). The user sends \( h' = h / a \mod q \) to signer \( U \).
3. Upon receiving \( h' \), signer \( U \) computes the quantity \( S' = r'(hinf + 1)Q_U + h'X_U \) and sends \( S' \) back to the user.
4. The user computes \( S = aS' + (hinf + 1)Q_U \). Therefore, \( (R, S) \) is signer \( U \)'s partially-blind signature on the blind message \( m \) and agreed common information \( info \).

When a receiver receives a partially-blind signed message, \( (ID_U, info, m, R, S) \), he or she verifies its validity according to the following steps:
1. Compute \( h = H_2(m, R), hinf = H_2(info) \), and \( Q_U = H_1(ID_U) \in G_1 \).
2. Compute the quantities \( A = e(S, P) \in G_2 \) and \( B = e(Q_U, (hinf + 1)R + hPpub) \in G_2 \). If \( A = B \), then the partially-blind signed message is valid. Otherwise, it is invalid.

The correctness of verification is as follows.
\[
S = aS' + (hinf + 1)Q_U = ar'(hinf + 1)Q_U + ah'X_U + b(hinf + 1)Q_U = (hinf + 1)(ar'Q_U + bQ_U) + hxQ_U
\]
\[
e(S, P) = e((hinf + 1)(ar'Q_U + bQ_U) + hxQ_U, P) = e(Q_U, (hinf + 1)R + hPpub)
\]

F. Performance of Signature Schemes

The signature schemes proposed in Subsections II.C, II.D, and II.E are based on bilinear pairing. For a practical performance analysis, the following description are based on the settings of Type A pairings in pairing-based cryptography \([42]\). An elliptic curve over the finite field \( F_p \) (denoted by \( E(F_p) \)) is the set of all points (solutions) of the equation \( y^2 = x^3 + x \). \( p \) is a 512-bit prime number so that \( p = 3 \mod 4 \). Therefore \( E(F_p) \) consists of \( p + 1 \) points and \( E(F_p^2) \) consists of \((p + 1)^2 \) points. \( G_1 \) and \( G_2 \) are subgroups of \( E(F_p) \) and \( E(F_p^2) \) respectively. \( q \) is the order of \( G_1 \) and \( G_2 \). \( q \) is a 160-bit prime number and divides \( p + 1 \), namely \( |q| = 160 \) and \( q | (p + 1) \). The security level is consistent with Level 4 security during years 2013–2015, as proposed by Lenstra and Verheul \([41]\) and ECRYPT II recommendations \([23]\). For the year 2019, \(|q| \) will be 192 for the same security level. Other recommendations about security levels can be found in \([11, 46, 49]\). The hash function \( H_2(\cdot) \) maps a message to elements of cyclic group \( G_1 \). The hash function \( H_3(\cdot) \) can be, for example, one of the cryptographic hash functions SHA-224, SHA-256, SHA-384, or SHA-512 \([47-48]\); the bit lengths of the hash values are 224, 256, 384, and 512, respectively.

For the signature schemes proposed in Subsections II.C, II.D, and II.E, Table 1 lists the signature sizes, costs of signature verification, and generation. In determining the computational costs, the computations of hash function and point addition were excluded. In addition, \( S_{ID}, P_M, H_M \) denote operations of scalar multiplication, pairing, and hash to point, respectively.
The computational costs of pairing were about 3–4 times those of scalar multiplication [29-30]. Performing $S_M, P_M$, and $H_M$ on a hardware platform with a PIV 3-GHZ processor, 512-MB memory, and a Windows XP operating system showed that the timings for $S_M, P_M$, and $H_M$ were 6, 20, and 3 ms [30].

All of the signature schemes generated the same size of signatures and required almost the same amount of verification time. However, compared to the digital signature, the timings of blind signature and partially-blind signature expended about 2.5 times in generating signature. Table 1 summarizes the computational cost and signature sizes of signature schemes.

Table 1. Computational cost and signature sizes of signature schemes

<table>
<thead>
<tr>
<th>Signature scheme</th>
<th>Signature generation</th>
<th>Signature verification</th>
<th>Signature size</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID-based digital signature (Sec. II.C)</td>
<td>3 $S_M$</td>
<td>1 $S_M$ + 2 $P_M$ + 1 $H_M$</td>
<td>1024 bits</td>
</tr>
<tr>
<td>ID-based blind signature (Sec. II.D)</td>
<td>7 $S_M$</td>
<td>1 $S_M$ + 2 $P_M$ + 1 $H_M$</td>
<td>1024 bits</td>
</tr>
<tr>
<td>ID-based partially blind signature (Sec. II.E)</td>
<td>7 $S_M$</td>
<td>2 $S_M$ + 2 $P_M$ + 1 $H_M$</td>
<td>1024 bits</td>
</tr>
</tbody>
</table>

G. Communication Channel Provides Key Agreement

This subsection describes a key agreement protocol, called KAP1. To secure the communications, initiators, and responders (the communication parties) must compute a shared session key. Subsequently, both of them use this session key to encrypt and decrypt the exchanged messages. The following scheme [55] has been proved to satisfy the security properties: known session key security, forward secrecy, key compromise impersonation resilience, and unknown key share resilience [18]. Let $H(x): G_1 \rightarrow \{0, 1\}^l$ be a key derivation function, where $|l| = 160$ meets Level 4 security. Assume that the initiator is user $U$ and responder is user $V$.

1. The initiator, user $U$, chooses a random number $a \in R Z_q$. Compute $Q_U = H_1(IID_U) \in G_1$ and $T_U = aP$. User $U$ sends $T_U$ to user $V$.

2. The responder, user $V$, chooses a random number $b \in R Z_q$. Compute $Q_V = H_1(IID_V) \in G_1$ and $T_V = bP$. User $V$ sends $T_V$ to user $U$ and computes the shared secret $k_V = e(bQ_U, P_{pub})e(X_U, T_V)$.

3. $U$ computes the shared secret $k_U = e(aQ_V, P_{pub})e(X_U, T_V)$. Clearly, the quantities $k_U$ and $k_V$ are equal, namely $k_U = e(aQ_V, P_{pub})e(X_U, T_V) = e(Q_V, P)^ae(1, P)^b = e(Q_V, P)^a = e(Q_U, P)^b = e(Q_U, P)^{k_U}$. Based on the computational Diffie-Hellman problem [22], user $U$ and $V$ share a secret session key, $sk = H_2(k_U) = H_2(k_V)$. With this session key, they exchange information confidentially.

H. Communication Channel Provides Identity Privacy for Initiator

This subsection describes a key agreement protocol preserving initiator’s identity privacy, called KAP2. In some cases, the user who initiates communication might want to keep her or his identity secret. Cryptography researchers have published key agreement protocols [35, 40, 43, 60] that provide user identification and key exchange simultaneously, while protecting a user’s personal information. The following protocol (KAP2) utilizes the advantages of identity-based cryptosystems [8] to simplify the procedure of the session key agreement. Assume that the initiator is user $U$ and the responder is user $V$.

1. The initiator, user $U$, chooses a random number $k \in R Z_q$. Compute $Q_U = H_1(IID_U) \in G_1$, $Q_V = H_1(IID_V) \in G_1$, $K_U = kQ_U$, $K_V = kQ_V$. Compute session key $sk = H_3(sk_U) = H_3(e(X_U, K_U))$. User $U$ sends $K_U$ to user $V$.

2. The responder, user $V$, computes session key $sk = H_3(sk_V) = H_3(e(X_V, K_U))$. The quantities $sk_U = e(X_U, K_U) = e(Q_U, Q_V)^{k_U}$ and $sk_V = e(X_V, K_U) = e(Q_U, Q_V)^{k_V}$ are equal. Therefore, user $U$ and $V$ share a secret session key, $sk = H_3(sk_U) = H_3(sk_V)$. With this session key, they exchange information confidentially. Knowing the quantity $K_U = kQ_U$, user $V$ cannot determine the identity of the initiator. The proposed key agreement protocol provides privacy for the initiator.

III. THE PROPOSED TRANSFERABLE E-CASH PAYMENT SYSTEM BASED ON A PROFITABLE MODEL

Like the traditional e-cash payment system, the proposed transferable e-cash payment system based on a profitable model involves three entities: the consumer, merchant, and bank. However, the bank can play two roles involving issuing the e-cash, verifying it, or both. The conduct of payment is divided into three stages: the withdrawal stage, payment stage, and deposit stage. The withdrawal stage consists of withdrawing quasi-e-cash and exchanging quasi-e-cash for e-cash. Figure 1 shows the information flows among the entities. The flow labeled $w-q$ indicates the process of withdrawing quasi-e-cash, $w-e$ the process of exchanging quasi-e-cash for e-cash, $p$ the process of the payment stage, and $d$ the process of the deposit stage. The processes are detailed in the following.

![Figure 1](image-url)

**Figure 1.** The processing architecture of the proposed payment system

A. Withdrawal Stage

The generation of new e-cash involves two steps: withdrawing quasi-e-cash and exchanging quasi-e-cash for e-cash. First, the consumer and issuing bank interactively generate quasi-e-cash. The quasi-e-cash consists of one partially-blind signature and one blind signature. The rationale
for using schemes involving a blind signature lies in the need to ensure anonymity, untraceability, and unlinkability. When applying for quasi-e-cash, a customer designates a verification bank where the quasi-e-cash will be exchanged for e-cash.

1) Withdrawing quasi-e-cash
Figure 2 depicts the message flows of withdrawing quasi-e-cash. The following steps describe the details.

Step 1. A consumer with the identity $ID_C$ must communicate with the issuing bank $ID_{IB}$. $ID_C$ and $ID_{IB}$ follow the key agreement protocol (KAP1 described in Subsection II.G) to establish a communication channel with a key agreement. Finally, $ID_C$ and $ID_{IB}$ agree on a shared session key. With this session key, they can exchange information secretly. Both of them negotiate certain information (e.g., the verification bank, the amount of e-cash, the denomination of e-cash, and the expiry date). That information related to each issue of e-cash is called common information and is denoted by $info = \{\text{denomination, expiry date}\}$.

Step 2. Following the procedures stated in Subsection II.D, the consumer obtains a blind signature $\sigma_B = (ID_{IB}, ID_{VB}, m', R_{cer}, S_{cer})$. Similarly, the consumer obtains a partially-blind signature $\sigma_B = (ID_{IB}, info, m, R, S)$ using the scheme detailed in Subsection II.E. Therefore, the quasi-e-cash is $\{\sigma_B, \sigma_B\}$. While the consumer and issuing bank interactively generate these signatures, the consumer chooses verification bank $ID_{VB}$, selects the random strings $m$ and $m'$, and performs affine transformation to obtain the quantities $R, S, R_{cer}$, and $S_{cer}$.

Figure 2. Withdrawing quasi-e-cash

2) Exchanging quasi-e-cash for e-cash
$ID_{VB}$ is the identity of the verification bank. The consumer selects the verification bank in the process of withdrawing quasi-e-cash: the duties of the verification bank are to detect double spending and verify signatures. Each verification bank maintains two tables, quasi-e-cash table and e-cash table. Table 2 is the quasi-e-cash table which is used to determine whether quasi-e-cash has already been exchanged for e-cash. Table 3 is the e-cash table; the verification bank maintains this table to detect double spending and issue signatures (e-cash) to the payee. Figure 3 depicts the message flows of exchanging quasi-e-cash for e-cash. The following steps describe the details.

Step 1. Let consumer $ID_C$ be the owner of a quasi-e-cash = $\{\sigma_B, \sigma_B\}$ and the verification bank be $ID_{VB}$. To exchange quasi-e-cash for e-cash, $ID_C$ must communicate with the verification bank $ID_{VB}$, $ID_C$ and $ID_{VB}$ follow the key agreement protocol KAP2 (Subsection II.H) to establish a communication channel. The protocol KAP2 provides privacy protection for the initiator. Therefore, $ID_{VB}$ receives no information regarding the identity of the initiator (consumer $ID_C$). Then, $ID_C$ prepares a request message $Request-e-cash = \{\text{ quasi-e-cash, Next VB}\}$ and sends it to $ID_{VB}$. It should not be confused with the verification bank, (i.e., $ID_{VB}$ and Next VB). The $ID_{VB}$ specified in $\sigma_B$ is the verification bank responsible for exchanging quasi-e-cash for e-cash. The message “Next VB” is the information regarding the verification bank where the exchanged e-cash is verified when the owner spends it.

Step 2. When receiving the Request-e-cash, $ID_{VB}$ searches $\sigma_B$ and $\sigma_B$ in the quasi-e-cash table. If either $\sigma_B$ or $\sigma_B$ is found then $ID_{VB}$ rejects the request and terminates the session. Otherwise, $ID_{VB}$ stores $\{\sigma_B, \sigma_B, Timestamp\}$ in a quasi-e-cash table and verifies the validity of $\sigma_B$ and $\sigma_B$. A successful verification causes $ID_{VB}$ to add the information, $M = \{\sigma_B, Timestamp, Next VB\}$, and the signing on $M$. The signature on $M$, $\sigma_B = (R_{ VB}, S_{ VB})$, is a traditional signature as stated in Subsection II.C. Subsequently, the e-cash is $\{M, \sigma_B\}$. $ID_{VB}$ sends the initiator (requester $ID_C$) e-cash and stores it in the e-cash table.

Step 3. Upon receiving the e-cash sent back by $ID_{VB}$, $ID_C$ verifies whether it is valid. Successful validation completes the transaction of exchanging quasi-e-cash for e-cash. An instance of Table 2 is quasi-e-cash $1 = \{\sigma_{PB1}, \sigma_{B1}\}$ and quasi-e-cash $2 = \{\sigma_{PB2}, \sigma_{B2}\}$, where $\sigma_{PB1} = (ID_{IB}, info, m_1, R_1, S_1)$, $\sigma_{B1} = (ID_{IB}, ID_ciy, m', R_{cer}, S_{cer})$, $\sigma_{PB2} = (ID_{IB}, info, m_2, R_2, S_2)$, and $\sigma_{B2} = (ID_{IB}, ID_ciy, m'_2, R_{cer2}, S_{cer2})$. The verification bank for these two quasi-e-cash processes is $ID_ciy$, as indicated in $\sigma_{B1}$ and $\sigma_{B2}$. $Timestamp_1$ and $Timestamp_2$ are the timestamps when $ID_ciy$ verified quasi-e-cash $1$ and quasi-e-cash $2$, respectively.

Table 2. quasi-e-cash table

<table>
<thead>
<tr>
<th>Partially blind signature</th>
<th>Blind signature</th>
<th>Timestamp</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sigma_{PB1} = (ID_{IB}, info, m_1, R_1, S_1)$</td>
<td>$\sigma_{B1} = (ID_{IB}, ID_ciy, m', R_{cer}, S_{cer})$, $Timestamp_1$</td>
<td></td>
</tr>
<tr>
<td>$\sigma_{PB2} = (ID_{IB}, info, m_2, R_2, S_2)$</td>
<td>$\sigma_{B2} = (ID_{IB}, ID_ciy, m'<em>2, R</em>{cer2}, S_{cer2})$, $Timestamp_2$</td>
<td>...</td>
</tr>
</tbody>
</table>
Likewise, Table 3 demonstrates the contents of e-cash = {M, σVB} and e-cash = {M2, σC2}, where M1 = (σPB1, Timestamp1, IDCB), σC1 = (Rc1, Sc1), M2 = (σPB2, Timestamp2, IDCB), and σC2 = (Rc2, Sc2). Timestamp1 and Timestamp2 are the timestamps when IDC signs the messages M1 and M2 respectively. In this example, Timestamp1 = Timestamp1 and Timestamp2 = Timestamp2. Subsequently, the verification bank for e-cash is IDC and IDCB is the verification bank for e-cash.

<table>
<thead>
<tr>
<th>Partially blind signature</th>
<th>Signature of Verification bank</th>
<th>Timestamp</th>
<th>Next VB</th>
</tr>
</thead>
<tbody>
<tr>
<td>σPB1</td>
<td>σC1</td>
<td>Timestamp1</td>
<td>IDCB</td>
</tr>
<tr>
<td>σPB2</td>
<td>σC2</td>
<td>Timestamp2</td>
<td>IDCB</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

**Figure 3. Exchanging quasi-e-cash for e-cash**

### B. Payment Stage

When a consumer wishes to purchase an item, he or she first requests the price. After accepting the quote for the purchase, the consumer pays for the order using e-cash. Figure 4 shows the information flow between consumer, merchant, and verification bank. The process is as follows.

**Step 1.** Let the consumer IDc be the payer and the merchant IDM be the payee. To buy something, IDc must communicate with merchant IDM. In addition, IDc does not want to disclose her or his identity; IDc runs the KAP2 to establish an anonymous and secure communication channel with IDM. With this communication channel, IDC and IDM agree on specific information (e.g., merchandise and prices). Finally, IDc sends e-cash to IDM.

**Step 2.** When receiving e-cash, IDM identifies the verification bank IDVB which is specified in the e-cash. To verify the e-cash, IDM must communicate with the verification bank IDVB. In addition, IDM wants to remain unidentified, IDM executes the KAP2 to establish a communication channel with IDVB. Then, IDM chooses “Next VB” which is the verification bank where the e-cash will be verified when IDM spends it. IDM sends IDVB a request message, Request-verification = (e-cash, Next VB). Once again, the verification banks IDVB and Next VB require further description. The verification bank IDVB was specified by IDC when he or she obtained the e-cash. Subsequently, IDC wants to transfer the ownership of this e-cash to IDM. Therefore, IDM chooses “Next VB” to denote the verification bank of the transferred e-cash.

<table>
<thead>
<tr>
<th>Consumer</th>
<th>Merchant</th>
</tr>
</thead>
<tbody>
<tr>
<td>Communication channel provide initiator’s identity privacy</td>
<td></td>
</tr>
<tr>
<td>e-cash = {M, σVB}</td>
<td>e-cash</td>
</tr>
</tbody>
</table>

**Figure 4. Payment stage**

**Step 3.** When receiving Request-verification, IDVB searches σVB in the e-cash table. If there are several entries of σVB, the entry with the most recent Timestamp is chosen. Let last-time denote the quantity of Timestamp stored in the record of the searched σPB. In the case of no entry of σPB, last-time = 0. Also, let this-time denote the quantity of Timestamp specified in M. *This-time* is the timestamp when the e-cash has been verified by a specific verification bank, *last-time* is the timestamp when the e-cash has been verified by the current verification bank. Therefore, IDVB accepts the request of verification if this-time ≥ last-time. The condition this-time ≥ last-time indicates no double spending. In addition, IDVB verifies the validity of e-cash, (i.e., verifies σVB). A successful verification leads IDVB to update Timestamp, collecting the related information, M = {σVB, Timestamp, Next VB}, and signing on M. The signature on M, σVB = (Rm, Svb), is a traditional signature as stated in Subsection II.C. Subsequently, the e-cash is {M, σVB}. IDVB
sends the initiator (requester) e-cash and stores it in the e-cash table.

**Step 4.** Upon receiving the e-cash sent back by \( ID_{VB} \), \( ID_M \) verifies whether it is valid. Successful validation terminates the transaction of the request verification. Also, \( ID_M \) sends the merchandise and information regarding payment completion to \( ID_C \). This completes the payment stage.

<table>
<thead>
<tr>
<th>Partially blind</th>
<th>Signature of</th>
<th>Timestamp</th>
<th>Next VB</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \sigma_{PB1} )</td>
<td>( \sigma_{cen1} )</td>
<td>Timestamp_{11}</td>
<td>( ID_{cn} )</td>
</tr>
<tr>
<td>( \sigma_{PB2} )</td>
<td>( \sigma_{cen2} )</td>
<td>Timestamp_{12}</td>
<td>( ID_{cn} )</td>
</tr>
<tr>
<td>( \sigma_{PB3} )</td>
<td>( \sigma_{cen3} )</td>
<td>Timestamp_{13}</td>
<td>( ID_{cn} )</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Table 4 shows an instance of the e-cash table after completion of a payment stage. Assume that \( ID_C \) sends e-cash\( \_1 = \{ M_1, \sigma_{cyn1} \} \) to \( ID_{DB} \), where \( M_1 = (\sigma_{PB1}, \text{Timestamp}_{11}, ID_{DB}) \). By inspecting the contents of \( M_1 \), \( ID_M \) knows that \( ID_{cen} \) is the verification bank. \( ID_M \) chooses \( ID_{DB} \) as the next verification bank, composes Request-verification = \( \{ \text{e-cash}, ID_{DB} \} \), and sends this request message to \( ID_{DB} \). Because \( \sigma_{PB1} \) is already stored in the e-cash table maintained by \( ID_{DB} \), last-time = \( \text{Timestamp}_{11} \). The information of \( M_1 \) indicates that this-time = \( \text{Timestamp}_{12} \). Therefore, last-time and this-time satisfy the condition of no double spending. Then, \( ID_{DB} \) verifies the signature \( \sigma_{cyn1} \) (i.e., computes the hash values \( h_{cyn1} = H_2 (M_1, R_{cyn1}) \) and \( Q_{cyn1} = H_1 (ID_{cen}) \) bilinear pairings \( B_1 = e (S_{cen1}, P) \) and \( B_2 = e (Q_{cyn1} h_{cyn1} P_{pub} + R_{cyn1}) \) ). If \( B_1 = B_2 \), then \( \sigma_{cyn1} \) is a valid signature.

After successfully verifying that no double spending occurred and the signature is valid, \( ID_{DB} \) signs on the message \( M_3 = (\sigma_{PB1}, \text{Timestamp}_{12}, ID_{DB}) \), where \( \text{Timestamp}_{12} \) is the current timestamp. The signature on \( M_3, \sigma_{cyn3} = (R_{cyn3}, S_{cyn3}) \), is a traditional signature as stated in Subsection II.C. Subsequently, the e-cash is \( \{ M_3, \sigma_{cyn3} \} \). \( ID_C \) sends the initiator (requester \( ID_M \)) the e-cash and stores it in the e-cash table.

**C. Deposit Stage**

When the consumer or merchant wants to exchange e-cash for cash, they enter the deposit stage. Figure 5 shows the information flow between the consumer, verification bank, and issuing bank. The details of the procedure are as follows.

**Step 1.** Let consumer \( ID_C \) be the owner of an amount of e-cash who wants to exchange it for cash. \( ID_C \) must communicate with the verification bank \( ID_{VB} \) which is specified in the e-cash. Thus \( ID_C \) executes KAP1 to establish a secure communication channel with \( ID_{VB} \). \( ID_C \) determines her or his bank account, deposit account, to store money after completing deposit stage. Then \( ID_C \) integrates related information as a request message, Request-deposit = \{e-cash, ID_C, deposit account\}. Finally, \( ID_C \) sends \( ID_{VB} \) Request-deposit.

**Step 2.** When receiving Request-deposit, \( ID_{VB} \) verifies the e-cash almost by the same procedures as described in the payment stage. \( ID_{VB} \) tests for double spending and verifies the validity of the e-cash. After successful verification, \( ID_{VB} \) identifies the issuing bank \( ID_{IB} \), which is specified in the e-cash. The following processing is dependent on whether \( ID_{VB} \) is equal to \( ID_{IB} \).

**Case \( ID_{VB} = ID_{IB} \):** \( ID_{VB} \) sets timestamp = \( \infty \) and Next VB = \( ID_{IB} \), signs on message \( M \) = \{ \sigma_{PB}, \text{Timestamp}, Next VB \} and generates signature \( \sigma_{VB} = (R_{VB}, S_{VB}) \). Subsequently, the e-cash is \( \{ M, \sigma_{VB} \} \). \( ID_{VB} \) stores the e-cash in the e-cash table, saves money in the deposit account, and terminates the deposit stage.

**Case \( ID_{VB} \neq ID_{IB} \):** \( ID_{VB} \) sets timestamp to the current moment and Next VB to the issuing bank \( ID_{IB} \), signs on message \( M \) = \{ \sigma_{PB}, \text{Timestamp}, Next VB \}. The resulting signature on \( M \) is \( \sigma_{VB} = (R_{VB}, S_{VB}) \) and the e-cash is \( \{ M, \sigma_{VB} \} \). \( ID_{VB} \) stores the e-cash in the e-cash table and sends Request-deposit = \{e-cash, ID_C, deposit account\} to \( ID_{IB} \). Upon receiving Request-deposit, \( ID_{IB} \) executes the aforementioned procedures, (i.e., Case \( ID_{VB} = ID_{IB} \)).

The Timestamp in the e-cash is crucial in detecting double spending. If Timestamp = \( \infty \), it implies that the e-cash has been revoked by the issuing bank. Neither \( ID_{VB} \) nor \( ID_{IB} \) sends consumers or merchants e-cash with Timestamp = \( \infty \). Therefore, the condition this-time \( \geq \) last-time stated in the payment stage is sufficient to reject all e-cash that has been revoked.

**Figure 5. Deposit stage**

**IV. ANALYSIS OF SECURITY**

In e-cash security is pivotal because the perceived security influences people’s willingness to use e-cash. This section presents a security analysis of the proposed e-cash payment system. It is demonstrated that the proposed system satisfies the requirements regarding anonymity, untraceability, unlinkability, verifiability, unforgeability, exchange between different denominations, prevention of double spending, decentralized payment processing, profitable payment model, and environmental friendliness.
A. Anonymity

Quasi-e-cash consists of a partially-blind signature $\sigma_{PB}$ and a blind signature $\sigma_{B}$, where $\sigma_{PB} = (ID_{PB}, info, m, R, S)$ and $\sigma_{B} = (ID_{B}, ID_{PB}, m', R_{e}, S_{cer})$. It is possible for the issuing bank $ID_{B}$ to retrieve the partial information $(info, R', h', S')$ of each signature when generating $\sigma_{PB}$. However, it is difficult for $ID_{B}$ to determine the identity of the signature receiver using this partial information, because the receiver blinds the message using two random numbers, $a$ and $b$, before sending the message to $ID_{B}$ to be signed. Then $ID_{B}$ sends back the signature $S'$. The receiver also uses the two blind factors $(a, b)$ to blind the signature $S'$, i.e., $S = a \cdot S' + b(H_{3}(info) + 1)Q_{IB}$. As discussed in the document [23, 62], there are $q$ pairs of $(a, b)$ to transform any $S'$ into $S$. Therefore, without knowing the exact quantities of $(a, b)$, $ID_{B}$ cannot determine the relationship between $ID_{B}$’s signed document and the blinded signature, i.e., $(R', S')$ and $(R, S)$. Similar circumstances apply to $\sigma_{B}$.

While in the exchange quasi-e-cash for e-cash stage, the verification bank cannot identify the owner of the quasi-e-cash and e-cash because the communication protocol KAP2 ensures privacy for the initiator of communication. Similarly, in the payment stage, the merchant cannot identify the consumer and the verification bank cannot identify the merchant. Thus, the proposed transferrable e-cash satisfies the requirements regarding anonymity.

B. Untraceability and Unlinkability

Assume that consumer $ID_{C}$ withdraws quasi-e-cash from the issuing bank $ID_{B}$, completes the exchange quasi-e-cash for e-cash stage to obtain e-cash, pays e-cash to the merchant by completing the payment stage, and the merchant receives the transferred e-cash from the verification bank $ID_{VB}$. In this scenario, quasi-e-cash, e-cash, and e-cash contain the same information $\sigma_{PB}$, but do not violate the properties of untraceability and unlinkability.

Because of the KAP2 protocol, the identity regarding who initiated the transactions is protected. Thus it is difficult to derive the identity of the payer and infeasible to determine whether two payment transactions originated from the same payer. The proposed payment scheme thus satisfies the properties of untraceability and unlinkability.

C. Verifiability

E-cash involves a traditional signature on a message, i.e., e-cash = $\{M, \sigma_{B}\}$, $M = (\sigma_{PB}, Timestamp, ID_{cer}(VB), \sigma_{B} = (R_{PV}, S_{PB})$. As the verification bank should publish its verification information, it is clear that anybody can verify the legality of e-cash.

D. Unforgeability

E-cash is essentially generated by three signature schemes, namely traditional signature $\sigma_{PB}$, blind signature $\sigma_{B}$, and partially-blind signature $\sigma_{PB}$. In the Random Oracle Model and the hardness of Computational Diffie-Hellman problem, $\sigma_{PB}$ is existentially unforgeable under the adaptively chosen message attack. Through a similar treatment in [64], $\sigma_{B}$ and $\sigma_{PB}$ are also unforgeable.

E. Exchange between Different Denominations

The properties of transferability, divisibility, and re-combinability make Bitcoin [5, 45, 52] very successful in e-cash payment. In Bitcoin transactions, a coin can be divided into $10^8$ Bitcoins and several coins can be combined to form a new coin. This implies that a coin might represent hundreds of Bitcoins or as low a value as $10^{-8}$ Bitcoins. Thanks to the partially-blind signature, the proposed e-cash payment system marks the denomination for each instance of e-cash. Consumers can withdraw e-cash with different denominations. Every verification bank has enough e-cash in different denominations. In executing the payment stage and ensuring payer = payee, the verification bank can break larger denominations of e-cash into several smaller denominations of e-cash and combine several smaller denominations of e-cash to form e-cash. Therefore, the properties of divisibility and re-combinability are achieved.

F. Prevention of Double Spending

In the exchanging quasi-e-cash for e-cash stage, the designated verification bank maintains the quasi-e-cash table to ensure every $\sigma_{PB}$ and $\sigma_{B}$ is exchanged for an e-cash once. A record of either $\sigma_{PB}$ or $\sigma_{B}$ indicates if the quasi-e-cash has been spent and is no longer useable. Similarly, in the payment and deposit stage, the designated verification bank maintains the e-cash table to prevent e-cash from being spent multiple times. The quantity last-time is the Timestamp stored in the record of the e-cash table and this-time is the Timestamp specified in the e-cash. The condition this-time $\geq$ last-time indicates the e-cash was verified by the current verification bank or another verification bank. Namely, the e-cash has been circulated in the e-cash payment system. The receiver of e-cash can only specify one verification bank; the timestamps related to the e-cash have been serialized. Therefore the condition this-time $\geq$ last-time is sufficient to prevent double spending.

G. Decentralized Payment processing

Based on preference, the receiver can choose the issuing bank to withdraw the quasi-e-cash and the verification bank to exchange or verify the e-cash. In addition, the issuing bank can also be the verification bank. Thus, the spread of computation and communication decreases server requirements.

H. Profitable Payment Model

The popular and widespread credit card system is accepted in most traditional business transactions. Crucial for the success of the credit card system is its profitable payment model. TTPs verify the legality of the credit card and receive a service fee and other marginal benefits. The proposed e-cash payment system is similar to the credit card system. The World Bank Alliance (WBA) plays the role of an e-cash payment system.
organizer. Any member of the WBA can issue and verify e-cash whereby any issuing bank can specify the denominations of the e-cash. Thus this e-cash payment system accepts every currency and is accepted by consumers around the world.

I. Environmental Friendliness

The proposed e-cash payment system saves energy compared with traditional e-cash payment systems. The reason is that the computational requirements of issuing new e-cash are higher than those required for verifying e-cash. The proposed payment system reuses e-cash based on the concept of transferability and thus reduces the requirement regarding computation. The performance analysis (in Section 5) provides further details.

V. PERFORMANCE ANALYSIS

To enable comparison, the proposed transferable e-cash payment system based on a profitable model is simply called transferable e-cash payment system. In the non-transferable e-cash payment system, the consumer receives e-cash by completing the withdrawal stage. The withdrawal stage involves generating and verifying a partially-blind signature. However, in the transferable e-cash payment system, the consumer receives e-cash by completing two stages, namely withdrawing quasi-e-cash and exchanging quasi-e-cash for e-cash. The comparisons of the computational costs are listed in Table 5. The cost for the transferable e-cash payment system is 24 $S_M + 10 P_M$. This is much higher than the cost incurred by the non-transferable e-cash payment system, which is $9 S_M + 2 P_M$.

Based on a profitable payment model, the economic incentive system is attractive to banks and TTPs, because by issuing and verifying e-cash, they can receive benefits. This aspect is consistent with the basic concept of capitalism.

VI. CONCLUSION

This study developed a transferable e-cash payment system based on a profitable model. The system exhibits the properties of transferability, anonymity, untraceability, unlinkability, verifiability, unforgeability, exchange between different denominations, prevention of double spending, decentralized payment processing, profitable payment model, and environmental friendliness. It is similar to an Internet-based cash payment system, except that e-cash is used rather than cash.

Based on a profitable payment model, the economic incentive system is attractive to banks and TTPs, because by issuing and verifying e-cash, they can receive benefits. This aspect is consistent with the basic concept of capitalism.

The proposed system is based on a modular design. The schemes involving traditional signatures, blind signatures, partially-blind signatures, and building communication channels are all replaceable. Thus, the issuing and verifying banks can freely choose their preferred cryptosystem. This feature provides banks with flexibility and confidence regarding security [63]. It also increases their willingness to join the payment system.

<table>
<thead>
<tr>
<th>Table 5: Computational cost of withdrawal e-cash and payment</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Transferable e-cash</strong></td>
</tr>
<tr>
<td>-------------------------</td>
</tr>
<tr>
<td><strong>Withdrawal stage</strong></td>
</tr>
<tr>
<td>Compute $G\sigma_B + V\sigma_B$</td>
</tr>
<tr>
<td>Cost $9 S_M + 2 P_M$</td>
</tr>
</tbody>
</table>

**REFERENCE**


