Off-line Digital Cash Schemes Providing Unlinkability, Anonymity and Change

Lynn Batten[1]and Xun Yi[2]

[1]*School of Information Technology, Deakin University, Australia*
[2]*School of Computer Science and Software Engineering, RMIT University, Melbourne, Australia*

Abstract

Several ecash systems have been proposed in the last twenty years or so, each offering features similar to real cash. One feature which to date has not been provided is that of a payee giving change to a payer for an e-coin in an off-line setting. In this paper, we indicate how an off-line ecash system can solve the change-giving problem. In addition, our protocol offers the usual expected features of anonymity and unlinkability of the payer, but can reveal the identity of an individual who illegally tries to spend ecash twice.

Keywords: Digital money, group blind signature, double spending

1. Introduction

Computer-based technology is significantly impacting our ability to access, store and distribute information. Among the most important uses of this technology is electronic commerce: performing financial transactions via electronic information exchanged over telecommunications lines. In particular, the development of secure and efficient ecash systems has been given a great deal of attention in recent years [4], [5], [10], [11].

Electronic commerce systems come in many forms including digital checks, debit cards, credit cards, and stored value cards. The usual security features for such systems are unforgeability, customer anonymity, and detection of double spending by the customer.

The type of electronic payment system we focus on here is ecash. Ecash (also called digital cash or ecash) is a term that is still not well defined in the research literature; rather, attempts to define it have been by listing desired

characteristics. As the name implies, ecash scheme is an electronic payment system modeled after the existing paper cash system. Paper cash has such features as: it is portable (easily carried), recognizable (as legal tender) hence readily acceptable, transferable (without involvement of a financial network), untraceable (no record of where money is spent), anonymous (no record of who spent the money) and it has the ability to make "change". Most designers of ecash focused on preserving the features of anonymity and unlinkability. Thus, ecash scheme is usually thought of as an electronic payment system that offers these two properties.

In improving electronic commerce take-up, trust has been shown to be a major factor [33]. The use of ecash online with only a limited value at risk is therefore preferable to the use of credit cards which can have very high credit limits. Anonymity in electronic transactions has also been shown to be a promoter of trust in the e-commerce environment and therefore beneficial to the online economy [38][2]. In recent years, the number of businesses accepting bitcoin continues to increase. Bitcoin is often described as an anonymous currency because it is possible to send and receive bitcoins without giving any personally identifying information.

In general, a payment system involves three parties, viz. banks, customers and shops, and three protocols, viz. withdrawal, payment and deposit (e.g. [26][44][45]). Execution of these three protocols in sequence constitutes the life-cycle of ecash, in which cash "travels" from a bank to a customer, from that customer to a shop, and back again from that shop to a bank.

A natural requirement for ecash systems is that the payment should be off-line, that is, not involving on-line cooperation with the bank; this ensures that purchases can be completed quickly without waiting for a bank (whose computer system may be down or loaded with similar enquiries) to process the information. Kane in [26] and Solat in [43] discuss these situations. Thus we distinguish between on-line and off-line ecash systems, depending on whether the bank needs or does not need to verify ecash during a transaction in real-time.

Off-line ecash systems are also highly preferable over on-line systems in case of low-value payments. In off-line ecash systems, the payee can accumulate payments, and deposit the aggregate value at the bank at suitable times when network traffic is low.

A fairly large body of recent cryptographic literature is devoted to the design of privacy-protecting off-line ecash systems. However, as we point out in the next section, no existing off-line ecash system can offer change.

In this paper, we present an off-line digital cash scheme which provides unlinkability, customer anonymity and the ability for a shop to give change. We enable the giving of the change feature by means of group blind signatures.

Related Work: The customer anonymity and unlinkability of ecash is usually achieved with a blind signature protocol, first introduced by Chaum [8] [9]. A blind signature allows a person to get a message signed by another party without revealing any information about the message to that party. Some electronic cash systems with blind signature have the property that an individual's spending cannot be determined even if all parties collude [11].

Forging paper cash is difficult. So, ideally, the illegal creation, copying, and re-use of ecash should be unconditionally or computationally impossible. Unfortunately, ecash is just digital information and is easily copied. Thus, rather than attempt to prevent copying of ecash, the focus has moved to detection of and response to copying. To protect against double spending, a bank needs to maintain a database of spent ecash and the deposit of cash arriving a second time should be rejected. In particular, in an off-line system, the best we can do is to detect when double spending has occurred, rather than prevent it. In this case, in order to protect the payee, it is then necessary to identify the payer which requires the disabling of the anonymity mechanism. This is at the heart of the subject-matter: it should be ensured that each ecash satisfies the property "once spent, concealed; twice spent, revealed" and referred to in this paper as 'protection against double spending', identified in [21], interpreted as meaning that the identity of a payer remains concealed if she does not act illegally but is revealed to the bank if she attempts to spend the same ecash twice. In order to accomplish this, the identity of the customer must be included in the cash somehow. This basic property of privacy-protecting off-line payment systems was first introduced in [10] as "accountability after the fact". There have been a number of papers in recent years offering a solution to this [3], [7], [18], [19], [25], [44]. Almost all of these off-line systems offer traceability as a security measure against double-spending based on a paradigm of Chaum, Fiat and Naor [10]. However, they still have to face several difficulties.

The first difficulty we mention is that the bank does not know what it is signing in the withdrawal step in accordance with the original meaning of blind signature. This introduces the possibility that the bank might be signing something other than what it is intending to sign. The existing

solution to this consists of specifying that the bank's digital signature be valid only in authorizing a withdrawal of a fixed amount. So, for example, the bank could have one key for a $10 withdrawal, another for a $50 withdrawal, and so on as described in [32]. This solution increases the complexity of key management of the ecash system.

The second difficulty we mention is that none of the off-line ecash systems offer the ability to make "change". (A recent paper by Kane [26] has a very flexible change protocol, but it must operate on-line with the bank.) For example, suppose that Alice is enrolled in a non-transferable, off-line cash system, and she wants to purchase an item from Bob that costs, say, $4.99. If she happens to have electronic coins whose values add up to exactly $4.99 then she simply spends these coins. However, it is unlikely that she will have the exact change for most purchases. Nor would she wish to keep a large reserve of coins on hand for some of the same reasons that one does not carry around a large amount of paper cash: loss of interest and fear of the cash being stolen or lost. Instead of change, the current solution as given in [31], [17], [32] and [7] to Alice's dilemma is to use divisible coins: coins that can be "divided" into pieces whose total value is equal to the value of the original coin. However, in these papers, this "division" of the coin needs to be done at the time the e-coin is constructed and not all possible purchase price configurations can be anticipated or accommodated.

A third difficulty we mention is that in all existing offline systems, the shop merchant sees the total value of the ecash proffered for payment. This is equivalent, in the paper cash world, to showing the merchant what bills are in your wallet before drawing out one with which to pay. We argue that this is unacceptable; a shop merchant presented with a large amount of ecash may be tempted to cheat the payer during or after the transaction.

Some existing ecash systems are surveyed as follows.

Early in 1991, Okomoto and Ohta [31] gave a divisible ecash scheme whose efficiency was improved by Eng and Okomoto in [17]. And in [32], Okomoto improved this scheme yet further. All these schemes depend on a tree structure to provide the divisibility feature.

Canard and Gouget [7] presented a construction of divisible ecash which generates a binary tree of keys. Their scheme is off-line, anonymous, unforgeable and discloses a double-spender.

Tan introduced an off-line ecash scheme based on blind proxy signatures from bilinear pairings [44]. The scheme permits anonymity of the user, with identification if the user double spends. The facility of giving change is not

available with this scheme.

In [3], Au, Susilo and Mu presented a zero-knowledge argument system, and, based on it, built an off-line ecash scheme with anonymity which traces double spending.

Juang's off-line scheme [25] used bilinear pairings over elliptic curves claiming that this reduces computational cost (see Table 1). Like the other schemes described here, it provides anonymity along with the ability to identify a double-spender.

Eslami and Talebi [18] proposed an off-line scheme which is anonymous, detects double-spending, equips the e-coins with an expiration date and enables portability of coins between storage devices. Blind signatures based on the ElGamal signature scheme are used to construct this model and it is proved to be secure against a number of types of fraud.

Everaere et al. [19] argued that while on-line payment environments are not efficient, they provide better protection against double-spending than do off-line environments. They take a novel risk management approach, allowing trade-offs between efficiency and effectiveness. No claim of e-coin unforgeability is made in the paper, and the authors of the present paper were unable to establish this property for [19].

In [22], the author constructs efficient blind and partially blind signature schemes over bilinear groups in the standard model. The main aim is efficiency in deriving signatures, and the protocols, aimed at 80-bit security, yield signatures consisting of only 40 bytes which is approximately 70% shorter than those in existing schemes with the same security in the standard model. This is achieved by restricting the signature derivation rounds to two moves. The author describes how these protocols compare favourably in every efficiency measure to all existing counterparts offering the same security in the standard model and even to many existing schemes in the random oracle model. While mention is made of applications to ecash, no ecash scheme is developed; however, the paper mentions that the unforgeability of the blindness protocols is based on intractability assumptions under the generic group model, and that their blindness holds with respect to malicious signing keys in the information-theoretic sense.

Finally, we mention the paper [40], which describes implementations of ecash systems. The authors provide their own model of an ecash scheme based on mobile trusted module architectures in which at design time, user payment tokens are composed of two modules: an untrusted but powerful execution platform (such as a smartphone) and a trusted but constrained

platform (referred to as a secure element). Their scheme fulfills all common properties of ecash while relaxing the amount of trust usually placed in the payment token. Moreover, they provide a proof-of-concept implementation using commercially available platforms.

None of the above ecash systems is able to give change. However, Kane [26] has an interesting scheme based on continued fractions which is able to give change in an on-line environment; the bank must confirm the change during the transaction.

In recognition of these deficiencies of the existing ecash systems, in this paper, we propose a new electronic off-line cash scheme which offers the change feature available in paper cash and prevents a shop from seeing the total value of the ecash proffered for payment. In addition, our scheme provides the required customer anonymity and unlinkability features; it is composed of a total of seven protocols: group membership key extract, payment certificate issue, ecash withdrawal, payment, change, deposit and revocation. While it is built on the well-known Schoenmakers ecash protocol [41] and the Ramzan blind signature group protocol [37], it is, to our knowledge, the first off-line ecash scheme providing change to the payer.

Summary of our contribution:
Our ecash scheme

- *operates off-line*
- *provides unlinkability*
- *provides customer anonymity*
- *detects double-spending*
- *allows a shop to give change without knowing the total value of ecash proffered for payment.*
- *is the only off-line cash scheme we know of which gives change.*

Subsequent sections are arranged as follows. In Section 2 we provide the technical background for the new protocols. Section 3 gives an overview of our ecash scheme and defines the security model. Section 4 provides details of the components of the protocols. Sections 5 and 6 deal with their security and performance. Conclusions are drawn in the final section.

2. Preliminaries

Our scheme is based on the discrete logarithm problem and on group blind signatures. We give the background details in this section.

2.1. Definitions

Let G be a cyclic group of order n generated by some $g \in G$, and $a \in Z_n^*$, where $n = pq, p = 2P+1, q = 2Q+1$ and p, q, P, Q are all large primes, G is a cyclic subgroup of $Z_{p'}^*$ where the definitions following can be found in Sections 3.2 and 3.3 of [6], p' is a prime and $n|(p'-1)$, a has large multiplicative order modulo p and q.

Definition 1. The **discrete logarithm** of $y \in G$ to the base g is the smallest non-negative integer x satisfying

$$g^x = y.$$

It is a widely held assumption that G, g can be chosen such that the discrete logarithm equation is infeasible to solve for x.

Definition 2. The **double discrete logarithm** of $y \in G$ to the base g and a is the smallest non-negative integer x satisfying

$$g^{a^x} = y,$$

if such an x exists.

G, g, n and a can be chosen such that the double discrete logarithm equation is infeasible to solve for x.

Definition 3. An ***e*-th root of the discrete logarithm** of $y \in G$ to the base g is an integer x satisfying

$$g^{x^e} = y,$$

if such an x exists.

If the factorization of n is unknown, computing e-th roots in Z_n^* is infeasible.

Definition 4. An $(\ell+1)$-tuple $(c, s_1, s_2, \cdots, s_\ell) \in \{0,1\}^\ell \times Z_n^\ell$ satisfying

$$c = H_\ell(m, y, g, g^{s_1}y^{c_1}, g^{s_2}y^{c_2}, \cdots, g^{s_\ell}y^{c_\ell}),$$

where c_i is the i-th leftmost bit of c, H is a hash function and H_ℓ represents the first ℓ bits of the output of H, is a **signature of knowledge of the discrete logarithm** of $y \in G$ to the base g on a message m, with respect to security parameter ℓ, denoted as

$$SKLOG_\ell[\alpha|y = g^\alpha](m).$$

Based on the Schnorr signature scheme [42], if the signer does not know the discrete logarithm of y to the base g, then it is infeasible for him to construct the $\ell+1$ tuple $(c, s_1, s_2, \cdots, s_\ell)$ satisfying the above equation. We can think of Definition 4 as an interactive protocol in which the c_i $(1 \leq i \leq \ell)$ represent challenges and the hash function H serves to remove the interaction. If the prover knows α such that $\alpha = \log_g y$, he can randomly choose $r_1, r_2, \cdots, r_\ell$ from Z_n, hash $m, y, g, g^{r_1}, g^{r_2}, \cdots, g^{r_\ell}$ with H to obtain a random challenge c, and obtain $s_1, s_2, \cdots, s_\ell$ by setting

$$s_i = r_i - \alpha \cdot c_i (mod\ n).$$

Definition 5. A **signature of knowledge of a double discrete logarithm** of y to the base g and a, on message m with security parameter ℓ, denoted as $SKLOGLOG_\ell[\alpha|y = g^{a^\alpha}](m)$, is an $(\ell+1)$-tuple $(c, s_1, s_2, \cdots, s_\ell) \in \{0,1\}^\ell \times Z_n^{*\ell}$ satisfying the equation,

$$c = H_\ell(m, y, g, a, P_1, P_2, \cdots, P_\ell),$$

where $P_i = g^{a^{s_i}}$ if $c_i = 0$ and $P_i = y^{a^{s_i}}$ otherwise.

Based on the Camenisch and Stadler group signature scheme [6], $SKLOGLOG_\ell[\alpha|y = g^{a^\alpha}](m)$ can be computed only if a double discrete logarithm α of the group element $y \in G$ to the base $g \in G$ and $a \in Z_n^*$ is known. Knowing α, the signature can be computed as follows.

1. For $1 \leq i \leq \ell$, randomly choose r_i from Z_n^*.
2. Set $P_i = g^{a^{r_i}}$ and compute $c = H(m, y, g, a, P_1, P_2, \cdots, P_\ell)$.
3. Set $s_i = r_i$ if $c_i = 0$ and $s_i = r_i - \alpha$ otherwise.

Definition 6. A signature of knowledge of an *e*-th root of the discrete logarithm of y to the base g, a message m, with security parameter ℓ, denoted

as $SKROOTLOG_\ell[\alpha|y = g^{\alpha^e}](m)$, is an $(\ell+1)$-tuple $(c, s_1, s_2, \cdots, s_\ell) \in \{0,1\}^\ell \times Z_n^{*\ell}$ satisfying the equation,

$$c = H_\ell(m, y, g, a, P_1, P_2, \cdots, P_\ell),$$

where $P_i = g^{s_i^e}$ if $c_i = 0$ and $P_i = y^{s_i^e}$ otherwise.

Based on the Camenisch and Stadler group signature scheme [6], such a signature can only be computed if the e-th root of the discrete logarithm α of y to the base g is known. When α is known, we can construct $SKROOTLOG_\ell[\alpha|\ y = g^{\alpha^e}](m)$ as follows.

1. For $1 \leq i \leq \ell$, randomly choose r_i from Z_n^*.
2. Set $P_i = g^{r_i^e}$ and compute $c = H(m, y, g, a, P_1, P_2, \cdots, P_\ell)$.
3. Set $s_i = r_i$ if $c_i = 0$ and $s_i = r_i/\alpha$ otherwise.

2.2. Group Signature

In general, a group signature scheme consists of the following five procedures [6].

Setup - a probabilistic algorithm that generates the group's public key $\mathcal{Y}$ and a secret administration key $\mathcal{S}$ for the group manager.

Join - an interactive protocol between the group manager and the new group member Bob that produces Bob's secret key x and his membership certificate A.

Signing - an interactive protocol between group member Bob and an external user Alice which takes as input a message m from Alice, and a secret key x from Bob, and produces a signature s on m.

Verifying - an algorithm which on input $(m, s, \mathcal{Y})$, determines if s is a valid signature for the message m with respect to the group public key $\mathcal{Y}$.

Open - an algorithm which, on input $(m, s, \mathcal{S})$, determines the identity of the group member who issued the signature s on the message m.

According to page 47 of Ramzan's thesis [37], a group signature scheme must satisfy the following security requirements.

1. **Unforgeability:** Only group members can issue a valid signature on behalf of the entire group, i.e., only group members can issue signatures that are verifiable by the group public key.
2. **Conditional Signer Anonymity:** Anyone can easily check that a message/signature pair was signed by some group member, but only the group manager can determine which member issued the signature.
3. **Undeniable Signer Identity:** The group manager can always determine the identity of the group member who issued a valid signature. Moreover, he can also prove to some other entity (such as a judge) which member signed a given document. (This feature is also referred to as 'non-repudiation'.)
4. **Unlinkability:** Determining if two different signatures were computed by the same group member is computationally infeasible for everyone except by the group manager.

Ramzan also mentions security against framing and collusion attacks. However, our focus is on fundamental security features 1 through 4 above for our scheme and we do not consider any attacks on it in this paper. The security requirements of the group blind signature scheme are very similar to those of the group signature scheme. Our proposal also has the blindness property in the signature; that is:

5. **Blindness of Signature:** The signer is unable to view the messages he signs. Moreover, the signer should have no recollection of having signed a particular document even though he (or anyone else for that matter) can verify that the signature is indeed valid.

This Blindness property holds because it is held in the Ramzan scheme. We do not use property 5 as a comparison with other schemes, but we need the assumption of this property in order to prove Theorems 3, 4 and 5 in Section 5.

3. Our Ecash Model

Our system is intended to provide an on-line shopping service to customers purchasing a product which can be delivered on-line. In this scenario, a customer wants to purchase an electronic commodity, for example, software or a movie, from an on-line shop with electronic cash through a computing-capable device, such as a PC, notebook, or mobile phone, connected to the Internet.

While our scheme is based on that of Schoenmakers [41] in association with the Ramzan group blind signature [37], we have added several novel components which combine to provide our off-line cash system with the capability of giving change, as indicated at the end of Section 1.

3.1. Participants

Our system is built on an existing group blind signature scheme [37] and involves four parties, viz.,

1. Customers (C) - register with banks, withdraw ecash from banks, pay ecash to online shops, and get "change" from shops.
2. Shops (S) - register with banks, receive ecash from customers, make "change" to customers, and deposit ecash in banks.
3. Banks (B) - issue ecash to customers and deposit ecash for shops.

In order to introduce our scheme's ability to give change, unlike the authors of [37] and [41], we require a fourth party:

4. Payment Certificate Authority (PCA) - issues one-time payment certificates to customers.

We assume that all bank branches form a few bank groups, each of which has a group manager (GM). Each shop is required to join a bank group by opening a bank account. The group manager of a bank group is in charge of issuing membership keys to bank branches and shop members. In addition, each customer is required to open a bank account with a bank group.

In the following description, we consider one bank group only where our ecash system is illustrated in Figure 1. We can easily extend our system to multiple bank groups.

3.2. Ecash Denominations

As with real money, our ecash scheme has a number of denominations. For example, US currency exists in 12 denominations: \$0.01, \$0.05, \$0.10, \$0.25, \$0.50, \$1, \$2, \$5, \$10, \$20, \$50, and \$100. We assume there exist a fixed total of m different denominations for ecash in our scheme. For each denomination, we assume that each GM has a public and private key pair by which membership keys for group members are extracted. Therefore, each group member, whether a bank branch or a shop, has m membership keys for the m different denominations of ecash. ecash with any value other than one of these m denominations can usually be represented by a sum of the m elementary electronic cash denominations.

3.3. Protocols

In general, any ecash system involves three basic protocols - withdrawal, payment and deposit (eg. [10]). ***Since the ability for a shop to give change to a customer will be introduced into our system, we require three additional protocols***

- membership key extract
- payment certificate issuing
- giving change.

Finally, we add a revocation protocol which revokes the public keys of anyone detected as spending an ecash twice. A brief description of the entire set of seven protocols needed is as follows; the starred ones (numbers 1, 2, 5 and 7) are new and specifically designed for our scheme.

*1 Membership Key Extract Protocol: A new member of a bank group (either a bank branch B or a shop S) interacts with the group manager (GM), which extracts secret m membership keys from the public key of the new member. Each membership key corresponds to one denomination of ecash. All these keys are sent to the new member through a secure channel and are unknown to other group members. The procedure for membership application is illustrated in Figure 1.

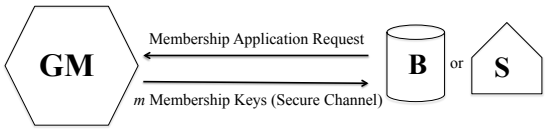

Figure 1: Membership Application

*2 Payment Certificate Issuing Protocol: To spend ecash, a customer (C), interacting with the Payment Certificate Authority (PCA), has his identification (embedded by both C and PCA) and one-time payment public key blindly signed by the PCA. In this way, the customer C can obtain multiple one-time payment certificates ($OTPC$) from the PCA, but the PCA knows nothing of these certificates. The procedure for payment certificate issuing is illustrated in Figure 2.

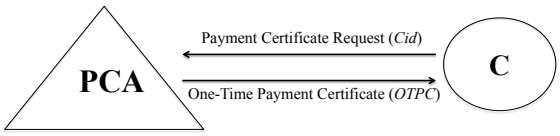

Figure 2: Payment Certificate Issuing

3 Withdrawal Protocol: To withdraw an ecash with a fixed denomination, a customer C, interacting with a bank branch B where the customer has a bank account, has his one-time payment certificate blindly signed by the bank branch B with the membership key corresponding to the denomination, on the basis of a group blind signature scheme. After the group blind signature is sent to the customer, the same amount of cash is deducted from the customer's account. The ecash is composed of the name of the bank group, the one-time payment certificate issued by the PCA, the denomination of the ecash, and the bank branch's group blind signature on the certificate. The procedure for e-cash withdrawal can be illustrated in Figure 3.

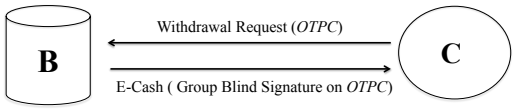

Figure 3: Withdrawal

4 Payment Protocol: To spend an ecash of a fixed denomination, a customer C, interacting with an online shop S from which the customer is purchasing an electronic commodity, sends to S the ecash together with his signature on the payment. The shop S accepts the payment if (i) the one-time payment public key is certified by the PCA; (ii) the group blind signature is produced by a member in a bank group; (iii) the signature on the payment is produced by a customer who knows the one-time payment secret key. The procedure for payment is illustrated in Figure 4.

*5 Change Protocol: To obtain change from an ecash with a fixed denomination, a customer C, interacting with an online shop S, has his new

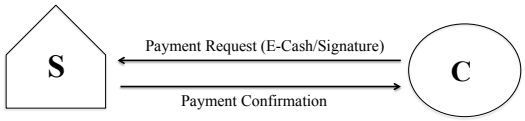

Figure 4: Payment

one-time payment certificate blindly signed by S, with the membership key corresponding to the denomination, on the basis of a group blind signature scheme. This protocol is the same as the withdrawal protocol except that the bank branch is replaced by the shop and the customer C is anonymous now. The procedure for change issuing is illustrated in Figure 5.

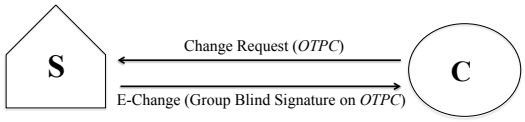

Figure 5: Change

6 Deposit Protocol: A shop S deposits a spent ecash paid by a customer C through the group manager GM of the specified bank group. The GM opens the identity of the signer of the group blind signature in the ecash and forwards the ecash to the corresponding bank branch B where the signer is either B itself or a shop S' having an account with B. The bank branch B checks the validity of the ecash as in the three points of the payment protocol and then transfers the value of the ecash to the shop S. In case the signer is a shop S', the bank branch deducts the same amount of cash from the account of the shop S'.

*7 Revocation Protocol: A bank group manager announces a list of revoked public keys of those group members who operate improperly.

3.4. Security Model

We define two of the main security features (unforgeability and customer anonymity) of our ecash scheme by means of games as described below.

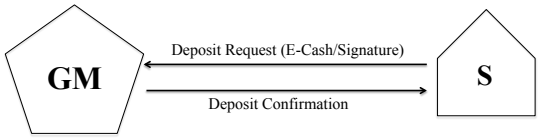

Figure 6: Deposit

(1) Customer anonymity

We define customer anonymity of an ecash system by means of a game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$, the latter who may be a bank, a shop or even the Payment Certificate Authority (PCA), as follows.

1. After initialisation of the ecash scheme, the adversary is provided with all public information, all public keys and all private keys of banks, shops and the Payment Certificate Authority (PCA). Note that we do not consider unforgeability here.
2. The adversary $\mathcal{A}$ chooses two customers C_0 and C_1 to challenge.
3. Each customer C_i $(i = 0, 1)$ played by the challenger $\mathcal{C}$ runs the payment certificate issuing protocol with the Payment Certificate Authorities (PCA) played by the adversary $\mathcal{A}$. In the end, each customer obtains a one-time payment certificate from the PCA.
4. Each customer C_i $(i = 0, 1)$ played by the challenger $\mathcal{C}$ runs the withdrawal protocol with a bank B played by the adversary $\mathcal{A}$ (or the change protocol with a shop S played by the adversary A). In the end, both customers obtain ecash with the same denomination from the bank (or the shop).
5. The challenger $\mathcal{C}$ randomly selects a customer C_b where $b = 0$ or 1. The customer C_b played by the challenger $\mathcal{C}$ runs the payment protocol with a shop S played by the adversary $\mathcal{A}$. In the end, the customer C_b spends the ecash in the shop S.
6. The challenger $\mathcal{C}$ provides the adversary $\mathcal{A}$ with all transcripts of the customer C_b in the payment protocol (i.e., all communication data from the customer C_b). The adversary $\mathcal{A}$ then outputs his guess b' where $b' \in \{0, 1\}$. If $b' = b$, the adversary $\mathcal{A}$ wins the game.

We define the advantage of the adversary $\mathcal{A}$ in breaking the customer anonymity of the ecash system as the probability that the adversary $\mathcal{A}$ guesses b correctly.

Definition 7. An ecash system has customer anonymity if the advantage of any Probabilistic Polynomial Time (PPT) adversary is not more than 1/2 plus any non-negligible value.

Remark. Since the adversary $\mathcal{A}$ may be a bank or a shop, he is able to examine the payment of the customer C_b in the deposit protocol in order to guess b.

(2) Unforgeability

Next, we define unforgeability for our ecash scheme. The adversary may be a customer, the PCA, a bank branch or a shop. When a bank branch or a shop is the adversary, we consider the case where the bank branch or the shop forges an ecash in the name of another bank branch or another shop.

Definition 8. An ecash system has unforgeability if any Probabilistic Polynomial Time (PPT) adversary, provided with ℓ ecash issued by a bank or a shop, where ℓ is the security parameter defined in Section 2, cannot produce a new ecash in the name of the bank or the shop with non-negligible probability.

4. Our Protocols

In this section, we first describe our setting and then give the details of our membership key extract, payment certificate issuing, withdrawal, payment, change and deposit protocols.

4.1. Setup

Assume that there are m denominations for ecash; this will assist in implementing our 'giving change' function. The trusted group manager of a bank group chooses a security parameter k and computes the following values:

1. m RSA public keys (N, e_i) and private keys d_i $(i = 1, 2, \cdots, m)$, where the length of N is at least $2k$ bits. We require that $N = pq$ where $p = 2p' + 1, q = 2q' + 1$ and p, q, p', q' are all primes. In practice, one should choose p and q to be significantly longer than k bits in order to ensure that factoring the modulus N is infeasible.
2. A cyclic group $\mathbb{G} =< \mathcal{G} >$ of order N for which computing discrete logarithms is hard. In particular, we choose $\mathbb{G}$ to be a cyclic subgroup of Z_P^* where P is a prime and $N|P - 1$. (See Subsection 2.1.)

3. An element $g \in Z_N^*$ where g has large multiplicative order modulo p and q.

The public key of the bank group is $GPK = \{N, e_1, e_2, \cdots, e_m, \mathbb{G}, g, \mathcal{G}\}$ while the secret key of the bank group is $GSK = \{d_1, d_2, \cdots, d_m\}$.

In addition, the trusted Payment Certificate Authority (PCA) chooses a security parameter l and computes the following values:

1. A cyclic group $\mathbb{H} = <h>$ of prime order n. One should choose n to be significantly longer than l bits in order to ensure that computing discrete logarithms is infeasible.
2. $y_1 = h^{x_1}, y_2 = h^{x_2}$ for randomly chosen x_1, x_2 from Z_n^*.

The public key of the PCA is $PCAPK = \{\mathbb{H}, h, y_1, y_2\}$ while the secret key of the PCA is $PCASK = \{x_1, x_2\}$.

We also assume the existence of a public-key infrastructure (PKI). The Payment Certificate Authority (PCA), any group manager (GM), any bank branch (B), any shop (S), and any customer (C) must be part of this PKI. All members of the PKI have public and private keys associated with this PKI, and know the public parameters of all parties.

4.2. Group Membership Key Extract Protocol

When a new member (either a bank branch B or a shop S) joins the bank group, it executes the following steps:

1. Picks up a randomly chosen secret key $x \in Z_N^*$.
2. Computes $y = g^x (mod\ N)$.
3. Signs y with its secret key associated with the PKI. This forces the new member to tie its identity with y.

To obtain a membership key, the new member sends y along with its signature on y to the group manager and proves that it knows x (without actually revealing x) using the signature of knowledge of the discrete logarithm. If the group manager is convinced that the new member knows x, he gives the new member m membership keys $v_1, v_2, \cdots, v_m$ for m denominations of ecash, where

$$v_i = (y+1)^{d_i} (mod\ N).$$

We assume that there exists a secure channel between the new member and the group manager during membership key extract.

4.3. Payment Certificate Issuing Protocol

To spend ecash, a customer C is required to obtain a one-time payment certificate from the PCA. Our payment certificate issuing protocol is developed based on Schoenmakers' withdrawal protocol [41] and is described as follows.

1. Using the identity c_{id} (for instance, the bank account number) of C, both PCA and C compute

$$h' = y_1^{c_{id}} y_2$$

2. PCA randomly chooses ω from Z_n^* and then computes

$$\alpha = h'^{\omega}$$

   Then PCA sends α to C.
3. C randomly chooses five integers t, t_1, t_2, μ, ν from Z_n^* and computes:

$$\begin{aligned} s_1 &= h'^{t} && (1)\\ s_2 &= y_1^{t_1} y_2^{t_2} && (2)\\ \alpha' &= \alpha h'^{\nu} h^{\mu} && (3)\\ \delta &= \mu + H(s_1, s_2, \alpha')(mod\ n) && (4) \end{aligned}$$

   where H is a secure hash function. Then C sends δ to the PCA.
4. PCA computes

$$\beta = \omega + \frac{\delta}{c_{id} x_1 + x_2}(mod\ n) \quad (5)$$

   and sends β to C.
5. C computes

$$\beta' = \frac{\beta + \nu}{t}(mod\ n) \quad (6)$$

   and verifies

$$s_1^{\beta'} = \alpha' h^{H(s_1, s_2, \alpha')} \quad (7)$$

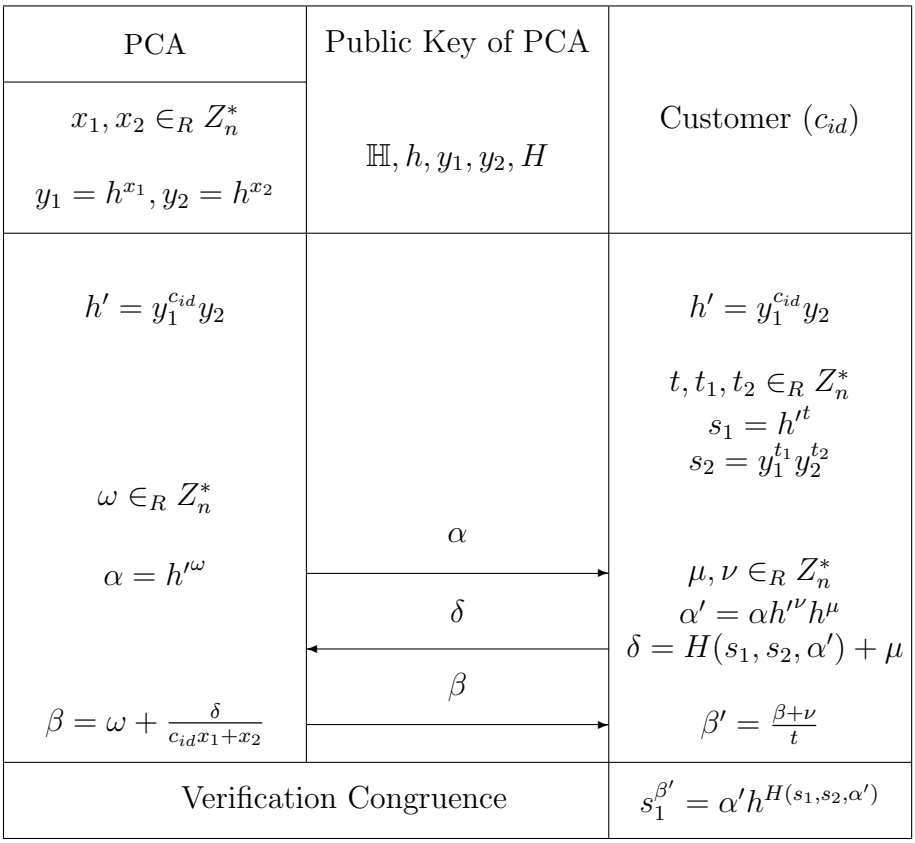

Figure 7: Payment Certificate Issuing Protocol where $\in_R$ indicates a random choice.

The one-time payment certificate is $OTPC = \{s_1, s_2, \alpha', \beta'\}$ where $(s_1, s_2, \alpha', \beta')$ satisfies Eq. (7) and the customer C keeps t, t_1, t_2 secret.

We assume that there exists a secure channel between C and PCA during payment certificate issuing. Our payment certificate issuing protocol is illustrated in Figure 7.

Remark. Each ecash is linked to a payment certificate of the customer. In order to prevent anyone from tracing the customer, a payment certificate should be used only once. This has the advantage that if a customer tries to spend the same ecash twice, he will be identified (see Theorem 2). It has the disadvantage that for each ecash withdrawn a new payment certificate must be issued.

4.4. Withdrawal Protocol

Our withdrawal protocol is based on the Ramzan group blind signature scheme in Chapter 4 of [37], where a customer C wants to obtain a group blind signature of a bank branch β, which belongs to a bank group B, on an ecash with a denomination λ. Assume that the customer C has obtained a one-time payment certificate $OTPC = \{s_1, s_2, \alpha', \beta'\}$ from the PCA, and that the bank branch β has obtained its group membership key v_λ extracted from its public key $y = g^x (mod\ N)$ such that $v_\lambda^{e_\lambda} = (y + 1)$. Our withdrawal protocol runs as follows.

1. The bank branch β randomly chooses r from Z_N^* and sets

$$\mathcal{G}' = \mathcal{G}^r$$

$$\mathcal{Z}' = \mathcal{G}'^y$$

   Then, as in [37], β chooses 2ℓ integers $r_{1,1}, r_{1,2}, \cdots, r_{1,\ell}, r_{2,1}, r_{2,2}, \cdots, r_{2,\ell}$ from Z_N^*, sets

$$P_{1,i} = \mathcal{G}'^{g^{r_{1,i}} (mod\ N)}$$

$$P_{2,i} = \mathcal{G}'^{r_{2,i}^{e_\lambda} (mod\ N)}$$

   for $i = 1, 2, \cdots, \ell$ (the security parameter defined in Subsection 2.1) and sends $P_{1,1}, \cdots, P_{1,\ell}, P_{2,1}, \cdots, P_{2,\ell}, \mathcal{G}', \mathcal{Z}'$ to the customer C.
2. The customer C randomly chooses two permutations $\pi_1, \pi_2 : \{1, 2, \cdots, \ell\} \rightarrow \{1, 2, \cdots, \ell\}$ and, as in [37], sets

$$Q_{1,i} = P_{1,\pi_1(i)}$$

$$Q_{2,i} = P_{2,\pi_2(i)}$$

   For $1 \leq i \leq \ell$, C randomly chooses 2ℓ integers $a_{1,i}, \cdots, a_{1,\ell}, a_{2,i}, \cdots, a_{2,\ell}$ and an integer w from Z_N^* and sets

$$\mathcal{G}'' = \mathcal{G}'^w$$

$$\mathcal{Z}'' = \mathcal{Z}'^w$$

$$R_{1,i} = Q_{1,i}^{wg^{a_{1,i}} (mod\ N)}$$

$$R_{2,i} = Q_{2,i}^{wa_{2,i}^{e_\lambda} (mod\ N)}$$

Then C computes

$$\begin{aligned} H_1 &= (c_{1,1}, c_{1,2}, \cdots, c_{1,\ell}) \\ &= H_\ell(s_1, s_2, \mathcal{Z}'', \mathcal{G}'', g, R_{1,1}, \cdots, R_{1,\ell}) \end{aligned}$$

$$\begin{aligned} H_2 &= (c_{2,1}, c_{2,2}, \cdots, c_{2,\ell}) \\ &= H_\ell(s_1, s_2, \mathcal{Z}''\mathcal{G}'', \mathcal{G}'', g, R_{2,1}, \cdots, R_{2,\ell}) \end{aligned}$$

where $H_\ell(\cdot) = H(\cdot)\|H(\cdot)$, and sends to β

$$\begin{aligned} H_1' &= (c_{1,1}', c_{1,2}', \cdots, c_{1,\ell}') \\ &= (c_{1,\pi_1(1)}, c_{1,\pi_1(2)}, \cdots, c_{1,\pi_1(\ell)}) \end{aligned} \tag{8}$$

$$\begin{aligned} H_2' &= (c_{2,1}', c_{2,2}', \cdots, c_{2,\ell}') \\ &= (c_{2,\pi_2(1)}, c_{2,\pi_2(2)}, \cdots, c_{2,\pi_2(\ell)}) \end{aligned} \tag{9}$$

3. For $1 \leq i \leq \ell$, the bank branch β computes

$$t_{1,i} = \begin{cases} r_{1,i} & \text{if } c_{1,i}' = 0 \\ r_{1,i} - x & \text{otherwise} \end{cases}$$

$$t_{2,i} = \begin{cases} r_{2,i} & \text{if } c_{2,i}' = 0 \\ r_{2,i}/v_\lambda (mod\ N) & \text{otherwise} \end{cases}$$

and sends $t_{1,1}, \cdots, t_{1,\ell}, t_{2,1}, \cdots, t_{2,\ell}$ to C.

4. The customer C verifies that

$$P_{1,i} = \begin{cases} \mathcal{G}'^{g^{t_{1,i}} (mod\ N)} & \text{if } c_{1,i}' = 0 \\ \mathcal{Z}'^{g^{t_{1,i}} (mod\ N)} & \text{otherwise} \end{cases}$$

$$P_{2,i} = \begin{cases} \mathcal{G}'^{t_{2,i}^{e_\lambda} (mod\ N)} & \text{if } c_{2,i}' = 0 \\ (\mathcal{Z}'\mathcal{G}')^{t_{2,i}^{e_\lambda} (mod\ N)} & \text{otherwise} \end{cases}$$

If verified, C computes

$$s_{1,i} = t_{1,\pi_1(i)} + a_{1,i}$$

$$s_{2,i} = t_{2,\pi_2(i)} a_{2,i}$$

for $i = 1, 2, \cdots, \ell$.

Finally, C obtains a group blind signature from the bank branch β on his one-time payment certificate, that is,

$$\mathcal{Z}'', \mathcal{G}'', \{H_1, s_{1,1}, s_{1,2}, \cdots, s_{1,\ell}\}, \{H_2, s_{2,1}, s_{2,2}, \cdots, s_{2,\ell}\}.$$

According to Definition 5, the $(\ell+1)$-tuple $(H_1, s_{1,1}, s_{1,2}, \cdots, s_{1,\ell})$ is a signature of knowledge of a double discrete logarithm of $\mathcal{Z}''$ to the base $\mathcal{G}''$ and g, on message (s_1, s_2), denoted as

$$SKLOGLOG[x|\mathcal{Z}'' = \mathcal{G}''^{g^x}](s_1, s_2)$$

According to Definition 6, $(\ell+1)$-tuple $(H_2, s_{2,1}, s_{2,2}, \cdots, s_{2,\ell})$ is a signature of knowledge of an e-th root of the discrete logarithm of $\mathcal{Z}''\mathcal{G}''$ to the base $\mathcal{G}''$ and g, on message (s_1, s_2), denoted as

$$SKROOTLOG[v_\lambda|\mathcal{Z}''\mathcal{G}'' = \mathcal{G}''^{v_\lambda^{e_\lambda}}](s_1, s_2)$$

An ecash takes a form of the denomination λ, the name of the bank group B, one-time payment certificate $OTPC = \{s_1, s_2, \alpha', \beta'\}$, the group blind signature $\mathcal{G}'', \mathcal{Z}''$, $V_1 = \{H_1, s_{1,1}, s_{1,2}, \cdots, s_{1,\ell}\}$, $V_2 = \{H_2, s_{2,1}, s_{2,2}, \cdots, s_{2,\ell}\}$. To verify the ecash, one can compute

$$P'_{1,i} = (\mathcal{G}''^{(1-c_{1,i})}\mathcal{Z}''^{c_{1,i}})^{g^{s_{1,i}(mod\ N)}}$$

$$P'_{2,i} = (\mathcal{Z}''^{c_{2,i}}\mathcal{G}'')^{s_{2,i}^{e_\lambda}(mod\ N)}$$

for $i = 1, 2, \cdots, \ell$ and then check that

$$H_1 = H_\ell(s_1, s_2, \mathcal{Z}'', \mathcal{G}'', g, P'_{1,1}, \cdots, P'_{1,\ell})$$

$$H_2 = H_\ell(s_1, s_2, \mathcal{Z}''\mathcal{G}'', \mathcal{G}'', g, P'_{2,1}, \cdots, P'_{2,\ell})$$

We assume that there exists a secure channel between the customer and the bank, such as SSL. The withdrawal protocol is illustrated in Figure 8.

4.5. Payment Protocol

At this point, the customer C can purchase an electronic commodity from a shop S on the Internet by paying with an ecash. The electronic commodity may be a subscription of an online service for certain period which can be accessed online with an account name and a password. We assume that an anonymity network (such as Tor [46] originally short for The Onion Router, a

Bank	Public Key	Customer
$x \in_R Z_n^*, y = g^x$ $v_\eta = (y+1)^{d_\eta}$	$N, e_1, e_2, \cdots, e_m, G, g, \mathcal{G}$	$OTPC = \{s_1, s_2, \alpha', \beta'\}$
$r, r_{1,i}, r_{2,i} \in_R Z_N^*$ $\mathcal{G}' = \mathcal{G}^r$ $\mathcal{Z}' = \mathcal{G}'^y$ $P_{1,i} = \mathcal{G}'^{g^{r_{1,i}}}$ $P_{2,i} = \mathcal{G}'^{r_{2,i}^{e_\lambda}}$	$\mathcal{G}', \mathcal{Z}'$ $P_{1,1}, \cdots, P_{1,\ell}, P_{2,1}, \cdots, P_{2,\ell}$ →	$a_{1,i}, a_{2,i}, w \in_R Z_N^*, \pi_1, \pi_2$ $Q_{1,i} = P_{1,\pi_1(i)}$ $Q_{2,i} = P_{2,\pi_2(i)}$ $\mathcal{G}'' = \mathcal{G}'^w$ $\mathcal{Z}'' = \mathcal{Z}'^w$ $R_{1,i} = Q_{1,i}^{wg^{a_{1,i}}}$ $R_{2,i} = Q_{2,i}^{wa_{2,i}^{e_\lambda}}$ $H_1 = H_\ell(s_1, s_2, \mathcal{Z}'', \mathcal{G}'', g, R_{1,1}, \cdots, R_{1,\ell})$ $H_2 = H_\ell(s_1, s_2, \mathcal{Z}''\mathcal{G}'', \mathcal{G}'', g, R_{2,1}, \cdots, R_{2,\ell})$
$t_{1,i} = r_{1,i} - xc'_{1,i}$ $t_{2,i} = r_{2,i}/v_\lambda^{c'_{2,i}}$	← $H'_1 = \pi_1(H_1), H'_2 = \pi_2(H_2)$ $t_{1,1}, \cdots, t_{1,\ell}, t_{2,1}, \cdots, t_{2,\ell}$ →	$s_{1,i} = t_{1,\pi_1(i)} + a_{1,i}$ $s_{2,i} = t_{2,\pi_1(i)} a_{2,i}$
Verification Congruence and Hash Values		$P'_{1,i} = (\mathcal{G}''^{(1-c_{1,i})} \mathcal{Z}''^{c_{1,i}})^{g^{s_{1,i}}}$ $P'_{2,i} = (\mathcal{Z}''^{c_{2,i}} \mathcal{G}'')^{s_{2,i}^{e_\lambda}}$ $H_1 = H_\ell(s_1, s_2, \mathcal{Z}'', \mathcal{G}'', g, P'_{1,1}, \cdots, P'_{1,\ell})$ $H_2 = H_\ell(s_1, s_2, \mathcal{Z}''\mathcal{G}'', \mathcal{G}'', g, P'_{2,1}, \cdots, P'_{2,\ell})$

Figure 8: Withdrawal Protocol where $\in_R$ indicates a random choice.

free software for enabling online anonymity), is available for the customer to send his order and payment information to the shop and receive the electronic commodity anonymously.

To purchase an electronic commodity from the shop S, the customer C submits to S (1) order information (OI) including the description of the product, the quantity and price of the product, the name of the shop, the purchase date and one-time public key used to protect the electronic commodity in the case that the electronic commodity is composed of an account name and a password ... ; (2) payment information (PI) including λ, B, $OTPC = \{s_1, s_2, \alpha', \beta'\}$, $\mathcal{G}'', \mathcal{Z}''$, $V_1 = \{H_1, s_{1,1}, s_{1,2}, \cdots, s_{1,\ell}\}$, $V_2 = \{H_2, s_{2,1}, s_{2,2}, \cdots, s_{2,\ell}\}$; and (3) a signature of C on OI and PI with his secret keys t, t_1, t_2, and which is generated as follows.

The customer C computes

$$d = H(H(OI)|H(PI)) \tag{10}$$

$$r_1 = c_{id} \cdot d \cdot t + t_1 \tag{11}$$

$$r_2 = d \cdot t + t_2 \tag{12}$$

where c_{id} is the identity of C. The signature (r_1, r_2) is valid if

$$y_1^{r_1} y_2^{r_2} = s_1^d \cdot s_2 \tag{13}$$

The shop S accepts the payment if (1) V_1 is a valid signature of the double discrete logarithm, i.e., $V_1 = SKLOGLOG[x|\mathcal{Z}'' = \mathcal{G}''^{g^x}](s_1, s_2)$;
(2) V_2 is a valid signature of knowledge of the root of the discrete logarithm, i.e., $V_2 = SKROOTLOG[v_\lambda|\mathcal{Z}''\mathcal{G}'' = \mathcal{G}''^{v_\lambda^{e_\lambda}}](s_1, s_2)$; (3) Eq. (7) holds; and (4) Eq. (13) holds.

The shop does not need the bank's help with authorising the payment from the customer as it can check the validity of the customer's payment directly by verifying the customer's one-time payment certificate, the group blind signature (issued by either a bank branch or a shop), and the customer's signature on the payment. If all are correct, the shop accepts the payment and sends the electronic product encrypted by the one-time public key specified in the order information to the customer via an anonymity network.

In case the customer C spends several ecash amounting to the price of the commodity, C and S need to run the payment protocol for each cash.

The payment protocol is illustrated in Figure 9.

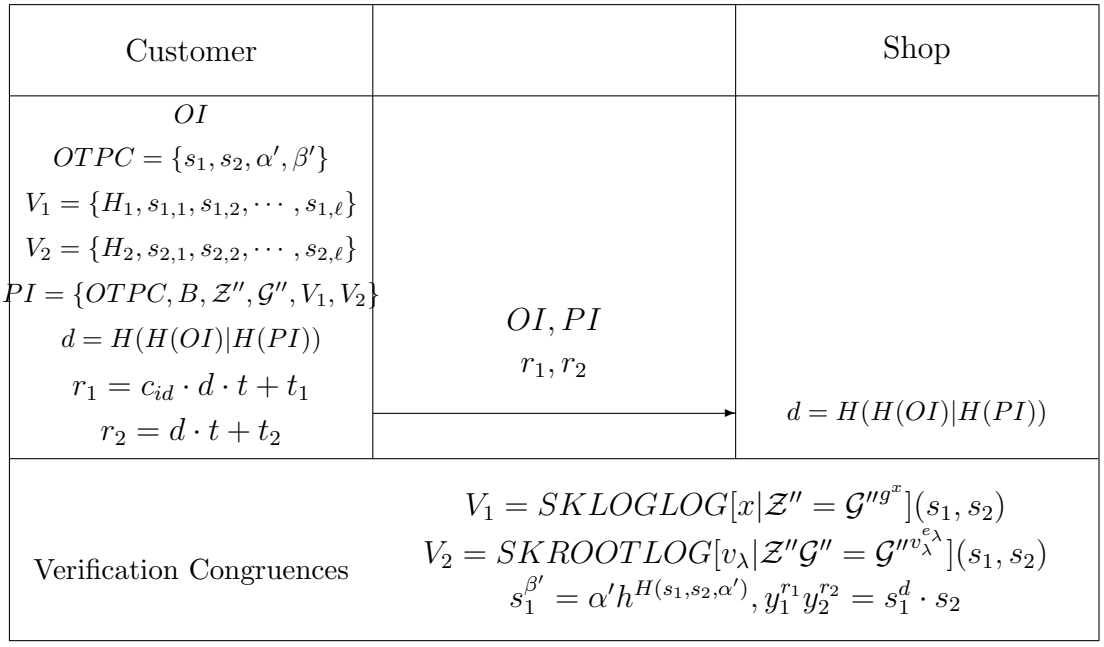

Figure 9: Payment Protocol

4.6. Change Protocol

We assume that only reputable shops, such as Safeway, Coles, Big W, Target, can join a bank group and obtain group membership keys. To make change of denomination λ, the customer C has his new one-time payment certificate blindly signed by the shop with the secret key x and the group membership key v_λ, in the same way as our withdrawal protocol, in which the role of a bank branch is played by the shop.

If the change is composed of several ecash, C and S need to run the change protocol several times.

4.7. Deposit Protocol

Our ecash system is built on a group blind signature scheme and our deposit protocol corresponds to the open procedure in the group blind signature scheme.

When a shop deposits an ecash λ, $H(OI)$, PI $(\{s_1, s_2, \alpha', \beta'\}, B, \mathcal{Z}'', \mathcal{G}'', \{H_1, s_{1,1}, s_{1,2}, \cdots, s_{1,\ell}\}, \{H_2, s_{2,1}, s_{2,2}, \cdots, s_{2,\ell}\})$ and (r_1, r_2) in the bank group specified in the payment information, the bank group manager (GM) searches for y such that

$$\mathcal{Z}'' = \mathcal{G}''^y \tag{14}$$

from the database (y). In this way, GM can reveal the identity of the signer of the group blind signature in the ecash. Based on the denomination, GM transfers the same amount of cash from the signer (either a bank branch or a shop) to the shop's account.

Double Spending Identification. In case two ecash with the same PI, i.e., $\{s_1, s_2, \alpha', \beta'\}$, $B, \mathcal{Z}'', \mathcal{G}''$, $\{H_1, s_{1,1}, s_{1,2}, \cdots, s_{1,\ell}\}$, $\{H_2, s_{2,1}, s_{2,2}, \cdots, s_{2,\ell}\}$ are deposited, double spending is detected as follows.

Assume that the rest of the two ecash details are $H(OI), PI, (r_1, r_2)$ and $H(OI'), PI, (r_1', r_2')$, respectively; we have

$$\begin{aligned} r_1 &= c_{id} \cdot d_1 \cdot t + t_1 \\ r_2 &= d_1 \cdot t + t_2 \\ r_1' &= c_{id} \cdot d_2 \cdot t + t_1 \\ r_2' &= d_2 \cdot t + t_2 \end{aligned}$$

where $d_1 = H(H(OI)|H(PI))$ and $d_2 = H(H(OI')|H(PI))$.

From the above equations, we can infer

$$r_1 - r_1' = c_{id} \cdot (d_1 - d_2) \cdot t \quad (15)$$
$$r_2 - r_2' = (d_1 - d_2) \cdot t \quad (16)$$

Since H is a secure hash function, it is computationally infeasible for the customer to construct two messages with the same hash value. Therefore, it is reasonable to assume $d_1 \neq d_2$. Thus the bank group manager GM can obtain the identity of the double spender by computing

$$c_{id} = \frac{r_1 - r_1'}{r_2 - r_2'} \quad (17)$$

GM asks the bank branch where the customer has an account to deduct the double spent cash from his account and transfer it into the account of the shop which deposited the ecash.

4.8. Revocation Protocol

To revoke a bank branch or a shop with a public key y, the bank group manager includes y in the list of revoked public keys and announces the list to the public. As in our deposit protocol, one can identify a revoked ecash by checking Eq. (14) given an ecash.

5. Security Analysis

This section analyses the security of our ecash system according to Definitions 7 and 8 specified in Section 3.

Theorem 1. *Assume that the Schoenmakers ecash system has customer anonymity and the Ramzan group blind signature scheme also has blindness (i.e., the signer is unable to view the messages he signs), and that there is an anonymity network for sending ecash to shops and for receiving electronic products. In this case, our ecash system has customer anonymity.*

Proof. According to the Customer Anonymity game specified in Section 3, in Step 6, the adversary is provided with all transcripts from the customer C_b, including (1) order information (OI) including the description of the product, the quantity and price of the product, the name of the shop, the purchase date and one-time public key ... ; (2) a payment information (PI) including λ (the denomination), B (the bank group), $OTPC = \{s_1, s_2, \alpha', \beta'\}$

(one-time payment certificate), $\mathcal{G}'', \mathcal{Z}''$, $V_1 = \{H_1, s_{1,1}, s_{1,2}, \cdots, s_{1,\ell}\}$, $V_2 = \{H_2, s_{2,1}, s_{2,2}, \cdots, s_{2,\ell}\}$ (the group blind signature of B on (s_1, s_2) with the secret key corresponding to denomination λ), and (3) (r_1, r_2) (a signature of C on OI and PI with his secret keys t, t_1, t_2).

As assumed in our security model, the PCA and the bank (or the shop) are all controlled by the adversary. To guess b, the adversary is able to examine each component of the transcripts from the customer C_b.

1. The OI does not contain any identification information of the customer.
2. The PCA adversary cannot distinguish customers C_0 and C_1 with $OTPC$ because the Schoenmakers withdrawal protocol by which we construct our Payment Certificate Issuing protocol is blind. (Note that if the Schoenmakers withdrawal protocol is not blind, the Schoenmakers ecash system does not have customer anonymity.)
3. The shop (or bank) adversary cannot distinguish customers C_0 and C_1 with $\mathcal{G}'', \mathcal{Z}''$, $V_1 = \{H_1, s_{1,1}, s_{1,2}, \cdots, s_{1,\ell}\}$, $V_2 = \{H_2, s_{2,1}, s_{2,2}, \cdots, s_{2,\ell}\}$ because the Ramzan group blind signature scheme used in our withdrawal protocol has blindness.
4. Given $(r_1, r_2), (s_1, s_2)$, the adversary cannot derive c (the identity of the customer C_b) from Eq. (10) - (13) because the Schoenmakers payment protocol by which we construct our payment protocol has customer anonymity.
5. The shop adversary cannot distinguish C_0 and C_1 by the source of the customer C_b, such as the IP address of the customer C_b, because the channel by which the shop receives the ecash from the customer C_b and delivers the electronic commodity to the customer C_b is an anonymous network.

The above analysis shows that the adversary cannot guess b correctly with a probability of more than $1/2$ plus a non-negligible value. Based on Definition 7, we conclude that our ecash system has anonymity and unlinkability. $\triangle$

Theorem 2. *Assume that the Ramzan group blind signature scheme has unforgeability; then our ecash system has unforgeability.*

Proof. The ecash takes the form of λ, B, $OTPC = \{s_1, s_2, \alpha', \beta'\}$, $\mathcal{G}'', \mathcal{Z}''$, $V_1 = \{H_1, s_{1,1}, s_{1,2}, \cdots, s_{1,\ell}\}$, $V_2 = \{H_2, s_{2,1}, s_{2,2}, \cdots, s_{2,\ell}\}$.

As assumed in our security model, the PCA is controlled by the adversary. Therefore, the adversary can construct any one-time payment certificate.

Each ecash contains a signature of the bank branch (or the shop), that is, $\mathcal{G}''$, $\mathcal{Z}''$, $V_1 = \{H_1, s_{1,1}, s_{1,2}, \cdots, s_{1,\ell}\}$, $V_2 = \{H_2, s_{2,1}, s_{2,2}, \cdots, s_{2,\ell}\}$.

If the adversary can produce a new ecash, he is able to forge a new signature of the bank branch (or the shop) which contradicts the assumption that the Ramzan group blind signature scheme has unforgeability. Therefore, Theorem 2 holds. $\triangle$

Our next three theorems provide proofs of double spending detection (Theorem 3) and unlinkability (Theorems 4 and 5).

Theorem 3. *Assume that the Schoenmakers ecash system satisfies "once concealed, twice revealed", and the Ramzan group satisfies Blindness of Signature, then our ecash detects double spending.*

Proof. Based on Theorem 1 and our assumptions, we know that our ecash system has customer anonymity.

Our ecash system identifies the double spender in the same way as the Schoenmakers ecash system. Based on the double spending identification described in Section 4.7, it can be seen that our ecash system can identify the double spender by computing $c = (r_1 - r_1')/(r_2 - r_2')$ and thus satisfies "twice revealed". $\triangle$

To further overcome double-spending, we can apply smart card techniques for the control of the customer's secret keys t, t_1, t_2 as in [20]. In this application, the bank issues a smart card, like a credit card, to the customer. The smart card generates secret keys t, t_1, t_2 for the customer during the payment certificate issuing. The secret keys are unaccessible to the customer. During the payment, the smart card ensures the secret keys are used to generate a signature only once.

In order to spend an ecash twice, the customer has to first compromise the smart card to obtain the secret keys t, t_1, t_2. This attack is very expensive.

Theorem 4. *Assume that the Schoenmakers ecash system has customer anonymity, the Ramzan group satisfies Blindness of Signature, and that there is an anonymity network for sending ecash to shops and for receiving electronic products. Then our ecash system satisfies unlinkability.*

Proof. Given two ecash spent by the same customer, an adversary, who may be the PCA, the bank or the shop, has three ways to link two ecash spent by the same customer.

1. The adversary links the two ecash according to their payment certificates. In our ecash system, each payment certificate is used by a cus-

tomer only once. The adversary first has to identify the customer from the payment certificates and then link the two ecash. Our payment certificate issuing protocol is built on the Schoenmakers withdrawal protocol, which is blind if the Schoenmakers ecash system provides customer anonymity. Therefore, this link attack can be prevented.

2. The adversary links the two ecash according to their signatures. Because the Ramzan signature is blind, this link attack can be prevented.
3. The adversary traces the source of the customer who sends the ecash, such as the IP address of the customer. If two ecash come from the same source, they are most likely spent by the same customer. Because the customer sends the ecash to the shop(s) over an anonymity network, this link attack can be prevented. △

Theorem 5. *Assume that the Ramzan signature satisfies Blindness of Signature, then determining if two different ecash were issued by the same group member (either a bank branch or a shop) is infeasible for everyone except for the bank group manager.*

Based on the properties of group blind signature, Theorem 5 can be proved with ease.

In [37], Ramzan presents an application of the blind signature idea to digital cash. There are three stages: withdrawal, spending and deposit, and when the shop deposits Alice's ecash at the bank, the bank is able to determine that it has not already been spent. By using blind signing, Alice remains anonymous both to the bank and the shop. As in our scheme, only the Central Bank is able to tell which bank branch issued the ecash. However, the shop is unable to provide change to Alice in the Ramzan scheme and so her ecash must be for the exact amount of the transaction. The distinguishing feature of the Ramzan scheme is argued to be its scaleability – shops and bank branches can join the Bank Group at any time. The scheme has a major flaw however – the shop must be online with the bank to verify the ecash before sending the sale item to Alice. The scheme we have developed in this paper uses the principal features of blind signature of the Ranzam scheme, and retains its scaleability, but in addition, it is able to function in an offline setting (the shop can issue goods before checking with the bank for double-spending) as well as adding the feature of providing change for an ecash larger than required.

Table 1 specifies the main similarities and differences between our scheme

and that of Ramzan [37].

Table 1. A comparison of our scheme with the Ramzan scheme.

Feature	**Ramzan [37]**	**Our Proposal**
Customer anonymity to shop and bank	YES	YES
The issuing bank branch is not identifiable	YES	YES
Double spending is detected	YES	YES
Scaleable (shops and bank branches can join any time)	YES	YES
Offline sales	NO	YES
Shop can give customer change	NO	YES

6. Performance Analysis

We analyse the performance of our ecash system and compare it with some existing ecash systems.

6.1. Changeability

Instead of giving change, some existing ecash systems [7], [17], [31], [32] use divisible coins: coins that can be "divided" into pieces whose total value is equal to the value of the original coin. However, this "division" of the coin needs to be done at the time the e-coin is constructed and not all possible purchase price configurations can be anticipated or accommodated.

Our proposed ecash system offers a simple solution to this problem. If the balance of the ecash provided by a customer is more than the price of the commodity that the customer wants to purchase, the shop can simply give the customer change. This change is a new ecash signed by the shop in the same way as our withdrawal protocol. If the change contains multiple denominations, the customer and the shop need to run the change protocol several times.

This change now acts as the new ecash for the customer instead of the old one. In addition, the change is certified with the group blind signature of the shop and the shop cannot deny the change.

In the deposit protocol, the bank manager is able to determine the identity of the shop which issued the change and then deduct the appropriate amount from the shop's account. If the shop issues change whose sum is more than the balance available to the shop, the bank can detect the malicious action and then execute effective measures against the malicious shop.

6.2. Computational Cost Comparison

As a measure of the practicality of our ecash system, we compare the computational cost of the customer in our system with four existing divisible ecash schemes . We follow the method of measurement used by [25]; thus, we assume that H is the computational time of one hashing operation (rather than a hash function), M is the computational time of one modular multiplication in a 1024-bit modulus, E is the computational time of one modular exponential operation in a 1024-bit modulus. Again following [25], we assume that $E \approx 240M$ and $H \approx 2/5M$. We do not include any measure of modular addition. The cost of inverting an element was calculated as for an exponentiation.

Table 2 shows the features of selected schemes and computes the cost of the three basic protocols needed in an ecash scheme: withdrawal, payment and deposit.

The authors of [17] suggest that a coin of value \$1000 would be sufficient for a person for several days, so we use the power of 2 counterpart $\$2^{10}$ as a benchmark to compare each of the papers in Table 2. In each case, the variable k is the logarithm base 2 of the size of the coin.

Table 2. Comparison of several ecash schemes

Properties	**Eng and Okamoto [17]1994**	**Okamoto [32]1995**	**Canard and Gouget [7] 2010**	**Tewari and Hughes [45] 2016**	**Our Proposal**
Unforgeability	Yes	Yes	Yes	Yes	Yes
Customer Anonymity	Yes	Yes	Yes	Yes	Yes
Detects double spending	Yes	Yes	Yes	Yes	Yes
Unlinkability	Yes	Yes	Yes	Yes	Yes
Operates off-line	Yes	Yes	Yes	No	Yes
Allows the giving of change	No	No	No	Yes	Yes
Computational cost to the users of withdrawal, payment and deposit	$19E + 15M$ $+(k+1)H$ $\approx 4575M$ $+kH$ $\approx 5394M$ (using $k \approx 2^{11}$)	$(4+2k)E+$ $2M+$ $(3+2k)H \approx$ $(963+481k)M$ $\approx 494,327M$ (using average $k \approx 2^{10}$)	$2^k(k+4)240M$ $+2^{k+1}H$ $+2^kM$ $\approx 7,373,211M$ (using $k \approx 2^{11}$)	$(2H+2E+M)2^{k-1}$ $+2^{(k-1)/2}M$ $\approx 482 * 2^{10}M$ $+32M$ $\approx 493,600M$ (where $k \approx 2^{10}$ is used as an average)	$0(k)E$ $\approx 240kM$ $\approx 15,360M$ ($k = 64$ if a 64-bit hash is used)

The first three columns of Table 2 present schemes which have a divisibility feature. All three of these schemes depend on tree structures and this

makes them costly.

In both [17] and [32] the nodes of a binary tree are used in the spending protocol. With root node worth 2^{10}, the entire tree has $(2^{11} - 1) = 2048$ nodes. In a worst case scenario, all of these nodes can be used. However, Okamoto states in [32] (page 446) that the *average* number of nodes used in this paper would be about half that. This reduces the computational cost of the payment protocol of this paper below that of [17], but as is shown in Table 2, the overall computational cost is much greater than that in [17]. The paper [7] again uses a binary tree in which the maximum height is referred to as L. In our evaluation we refer to L as k and set it at approximately 2^{11} for a fair comparison. The total computational cost is the worst of all papers.

In [45], Tewari and Hughes use a linear block chain method rather than a binary tree. A coin recipient needs to verify all transactions of previous owners. If the coin is worth $\$2^{10}$, then we will assume that the longest chain of transactions is $k = 2^{11}$ (which can be set as a maximum length before which the coin expires). The recipient of a coin must ensure that each transaction indicated in the coin is valid. The recipient must also solve a discrete logarithm problem modulo k, for which we have assumed $\sqrt{2^{10}} = 2^5$ multiplications, assuming an average chain length of 2^{10}.

Our performance compares favourably with the schemes of the first four columns. It is the only one that both gives change and operates off-line, and it is much more computationally efficient than all the others except [17].

7. Conclusion

With the development of electronic technology, computer networks, and modern cryptography, ecash is expected to replace paper money due to its high efficiency and security.

Until now, two trends in the research area of ecash have been visible. Starting with the introduction of payment schemes in the field of cryptography by Chaum, Fiat and Naor, research contributions have tended either to introduce new features into existing ecash paradigms or to address stronger attack models.

Our paper presents a new ecash system with the ability to give change. For instance, in the paper of Tewari and Hughes [45], the authors use the blind signature protocol of Chaum [8] along with Bitcoin protocols [30] to produce a fully anonymous and transferable ecash system which authenticates

transactions and prevents double-spending. They claim that their major contribution is a novel delegated signature scheme with distributed verification, which enables secure anonymous transfer of coins from one user to another without the need to contact a trusted third party. Like us, the authors assume that the bank has a number of coin denominations for each of which it provides a distinct signature key. Unlike our ability to transact off-line and our dependence on the bank to establish after-the-fact, that a coin has been double-spent, their double-spending and revealing of the double-spender is done within the block chain system and so is technically an 'on-line' detection. At about the same time, the authors of [49] assisted the development of stronger attack models by focusing on defining security and providing formal proofs of security of desirable ecash features in the standard model for their scheme, which includes use of dynamic group signatures. In order to place the ecash system on a mathematical foundation, we introduce the definitions of digital signature, hash function, certificate and the concepts of blind signature, group blind signature and blind certificate.

The proposed ecash system is composed of seven protocols: group membership key extract, payment certificate issue, ecash withdrawal, payment, change, deposit and revocation. In the certificate issue protocol, a customer obtains a blind certificate from the certificate authority. With the certificate, the customer exchanges some paper cash into an electronic token at a bank in the load protocol. When the customer purchases some commodity from an on-line shop, he pays the ecash to the shop following the payment protocol. An ecash is made using a combination of the blind certificate of the customer, the electronic token and the signature of the customer on this payment. If the amount of the electronic token is more than the price of the commodity, the shop carries out the change protocol to give the customer a new electronic token in "change". At a suitable time, the shop deposits the ecash paid by the customer at the specified bank on the basis of the deposit protocol. Finally, once any security leak relevant to a blind certificate (e.g., the secret signing transformation of the customer has been exposed or double-spending has been detected) appears, the revocation protocol can revoke any ecash based on the blind certificate.

The proposed ecash system satisfies the property "once concealed, twice revealed" and provides anonymity and unlinkability to customers. The security of the proposed system is ensured by a series of theorems.

Considering future work, we note that in the case of an on-line ecash system, a "change" to one payment can be given by a bank. This would

prevent a shop from knowing where a customer has shopped in the past. However, the proposed off-line ecash cannot satisfy this property because the change is offered by a shop instead of a bank. To overcome this weakness, in future work, we will investigate the possibility of a shop making use of a digital signature algorithm with one time pair of keys to anonymously issue an electronic token as"change" for customers. Although this kind of "change" no longer contains the identity of the shop, the bank can uniquely determine the identity of any shop by its certificate and signature.

Our future work will thus try to apply the digital signature algorithm with a one time pair of keys in the proposed ecash system and further optimize it.

Acknowledgement

The authors would like to thank Professor Yuliang Zheng for his valuable comments on the double discrete logarithm problem and Ms. Kalpana Singh for her comments on parts of earlier versions of this paper, and for her assistance in assembling Table 2.

[1] S.G. Akl, "Digital signature: A tutorial survey", IEEE Computer Magazine, pp.15-24, Feb. 1983.

[2] G. Antoniou and L.M. Batten, "E-commerce: protecting purchaser privacy to enforce trust", Journal of Electronic Commerce Research, vol. 11, pp. 421-456, 2011.

[3] M. Au, W. Susilo and Y. Mu, "Proof-of-knowledge of representation of committed value and its applications", LNCS 6168, pp. 352-369, 2010.

[4] S. Brands, "Untraceable off-line cash in a wallets with observers", Advances in Cryptology -Proceeding of Crypto'93, pp.302-318.

[5] S. Brands, "An efficient off-line electronic cash system based on representation problem", C.W.I. Technical Report CS-T9323, The Netherlands, 1991.

[6] J. Camenisch and M. Stadler, "Efficient group signature schemes for large groups (extended abstract)", in Proc. Crypto'97, 1997.

[7] S. Canard and A. Gouget, "Multiple denominations in ecash with compact transaction data", LNCS 6052, pp. 82-97, 2010.

[8] D. Chaum, "Blind signatures for untraceable payments", In Advances in Cryptology, Crypto '82, pp. 199-203, Springer-Verlag, 1983.

[9] D. Chaum, "Security without identification: transaction systems to make big brother obsolete", Communications of the ACM, 28(10): 1030-1044, October 1985.

[10] D. Chaum, A. Fiat and M. Naor, "Untraceable electronic cash", Advances in Cryptology -Crypto'88, LNCS 403, Springer-Verlag, pp.319-327.

[11] D. Chaum, B. Den Boer, E. Van Heijst, S. Mjolsnes and A. Steenbeen, "Efficient off-line electronic checks", Advances in Cryptology - Eurocrypt'89, LNCS 434, Springer-Verlag, pp.294-301.

[12] D.W. Davies and W.L. Price, "The application of digital signature based on public key cryptosystem", in Proc. 5th Internat. Conf. Computer Commun., Atlanta, GA, Oct. 27-30, 1980.

[13] D.W. Davies, "Applying the RSA digital signature to electronic mail", IEEE Computing Magazine, Vol.16, No.2, pp.55-62, Feb. 1983.

[14] D.E. Denning, "Protecting public keys and signature keys", IEEE Computer Magazine, Vol. 16, No. 2, pp. 27-35, 1983.

[15] W. Diffie and M.E. Hellman, "New direction in cryptography", IEEE Trans. Information Theory, Vol. IT-22, No.6, pp.644-654, Nov.1976.

[16] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithm", IEEE Trans. Inform. Theory, Vol. IT-31, pp. 469-472, 1985.

[17] T. Eng and T. Okamoto, "Single-term divisible electronic coins", Advances in Cryptology - Eurocrypt'94, Springer-Verlag, pp.311-323.

[18] Z. Eslami and M. Talebi, "A new untraceable off-line cash system", Electronic Commerce Research and Applications, vol. 10, pp. 59-99, 2011.

[19] P. Everaere, I. Simplot-Ryl and I. Traore, "Double spending protection for ecash based on risk management", LNCS 6531, pp. 394-408, 2011.

[20] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problem", in Lecture Notes in Computer Science 263, Advanced in Cryptology: Proc. Crypto'86, Aug. 11-15, 1986, pp. 186-194, Springer-Verlag, 1987.

[21] M. Franklin and M. Yung, "Secure and efficient off-line digital money", In Automata, Languages and Programming, ICALP'93, Berlin, 1993, LNCS 700, Springer-Verlag, pp.265-276.

[22] Ghadafi, E., Efficient Round-Optimal Blind Signatures in the Standard Model. Eprint iarc.org. (This is the full version of the extended abstract which appears in Proceedings of Financial Cryptography and Data Security 2017.)

[23] S. Goldwass, S. Micali and R.L. Rivest, "A 'paradoxical' solution to the signature problem", in Proc.25th Ann. IEEE Symp. Foundation Computer Science, Singer Island, FL. Oct.24-26, 1984.

[24] M. Jakobsson and A. Juels, "X-Cash: Executable digital cash", Proceeding of Financial Cryptography '98.

[25] W.-S. Juang, "RO-Cash: An efficient and practical recoverable pre-paid offline ecash scheme using bilinear pairings", The Journal of Systems and Software, vol. 83, pp. 638-645, 2010.

[26] A.M. Kane, "On the use of Continued Fractions for Electronic Cash", International Journal of Computer Science and Security, vol. 4, pp. 136-148, 2008.

[27] L.M. Kohnfelder, "A method for certification", MIT Laboratory for Computer Science, Cambridge, MA, May 1978.

[28] R.C. Merkle, "Secrecy, authentication and public key system", Ph.D. Thesis, Standford University 1979.

[29] C.J. Mitchell, F. Piper and P. Wild, "Digital Signature", Contemporary Cryptology - The Science of Information Integrity, IEEE Press, 325-378, 1992

[30] Nakamoto, S. "Bitcoin: A Peer-to-Peer Electronic Cash System", `http://www.bitcoin.org`, 2008

[31] T. Okamoto and K. Ohta, "Univeral Electronic Cash", Advanced in Cryptology - Crypto'91, Springer-Verlag, pp.324-337.

[32] T. Okamoto, "An efficient divisible electronic cash schemes", Advanced in Cryptology - Crypto'95, Springer-Verlag, pp.438-451.

[33] M.A. Patton and A. Josang, "Technologies for trust in electronic commerce", Journal of Electronic Commerce Research, vol. 4, pp. 9-21, 2004.

[34] S.C. Pohlig and M.E. Hellman, "An improved algorithm for computing logarithm over GF(p) and its cryptographic significance", IEEE Trans. Inform. Theory, Vol. IT-24, pp. 106-110, 1978.

[35] M.O. Rabin, "Digitalize signature", In Foundations of Secure Computing, New York: Academic, pp. 155-168, 1978.

[36] M.O. Rabin, "Digitalize signature and public-key functions as intractable as factorization", Massachusetts Institute of Technology Lab for Computer Science, Cambridge, MA, Technical Report, MIT/LCS/TR-212, Jan, 1979.

[37] Z.A. Ramzan, "*Group Blind Digital Signatures: Theory and Applications*", PhD Thesis, MIT, 1999.

[38] M. Rennhard, S. Rafaeli, L. Mathy, B. Plattner and D. Hutchison, "Towards pseudonymous e-commerce", Journal of Electronic Commerce Research, vol. 4, pp. 83-111, 2004.

[39] R.L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signature and public-key cryptosystems", Commun. ACM, Vol.21, pp. 120-126, 1978.

[40] Scheir, M., Balasch, J., Rial, A., Preneel, B. and Verbauwhede, I., 2015. Anonymous Split E-CashToward Mobile Anonymous Payments. *ACM Transactions on Embedded Computing Systems (TECS)*, 14(4), p.85.

[41] B. Schoenmakers, "An efficient electronic payment system withstanding parallel attacks", CWI Report CS-R9522, 1995.

[42] C.P. Schnorr, "Efficient identification and signatures for smart cards", in Proceedings of Crypto '89, pages 239-252, 1989.

[43] Solat, S., 2017. Security of Electronic Payment Systems: A Comprehensive Survey. *arXiv preprint arXiv:1701.04556*, 29 pages.

[44] Z. Tan, "An off-line electronic cash scheme based on proxy blind signature", The Computer Journal, vol. 54, pp. 505-512, 2011.

[45] Tewari, H. and Hughes, A., 2016. Fully Anonymous Transferable Ecash. IACR Cryptology ePrint Archive, 2016, 18 pages. Zhou, F., Li, Y., Zhou, Q., Miao, J. and Xu, J., 2016. The ecash system based on non-interactive zero-knowledge proofs. *International Journal of Computer Mathematics*, 93(2), pp.239-257.

[46] Tor (anonymity network), `http://en.wikipedia.org/wiki/Tor\_(anonymity\_network)`.

[47] U.S. Department of Commerce, National Institute of Standards and Technology, "A proposed federal information processing standard for digital signature standard (DSS)", Federal Register, August 30, 1991.

[48] L. Zhang, F. Zhang, B. Qin and S. Liu, "Provably-secure electronic cash based on certificateless partially-blind signatures", Electronic Commerce Research and Applications, vol. 10, pp. 545-552, 2011.

[49] Zhou, F., Li, Y., Zhou, Q., Miao, J. and Xu, J., 2016. The electronic cash system based on non-interactive zero-knowledge proofs. *International Journal of Computer Mathematics*, 93(2), pp.239-257.