Efficient Electronic Cash with Restricted Privacy

Cristian Radu, René Govaerts and Joos Vandewalle

Katholieke Universiteit Leuven, Laboratorium ESAT
Kardinaal Mercierlaan 94, B-3001 Heverlee, Belgium

Abstract. In this paper we propose a coin-based electronic payment system suitable for small payments. It is derived from Brands' scheme presented at Crypto'93, in the sense that the coins are built using the representation problem. The main contribution of our solution consists of the speedup of the withdrawal protocol. The gain of efficiency is achieved preserving the same level of integrity for user, shop and bank. A coin remains untraceable with respect to the user. This feature is fulfilled even if one assumes that the bank has unlimited computing power and colludes with shops in order to trace a coin to a specific user. However, a set of coins are linkable to a pseudonym of the user, restricting in this way his privacy. This drawback can be limited by "rotating" coins derived from different pseudonyms in a set of consecutive payment transactions.

1 Introduction

In the first off-line untraceable payment system proposed by Chaum, Fiat, and Naor [4], as well as in some subsequent papers [3, 10], the identification of double-spenders requires multiple terms in each coin. In these schemes, each term is used to answer one bit of challenge from the shop. The single-term coins, almost simultaneously proposed by Franklin and Yung [7, 8], Ferguson [6] and Brands [1, 2], provide a better efficiency and lower computational costs for coins. After the introduction of the "electronic wallet with observer" paradigm [5], the prior restraint of double-spending in combination with single-term coins became almost a standard approach in the design of privacy-protecting off-line electronic payment systems. As the observer will only cooperate in spending each coin once, the double-spending detection mechanism serves as a second level of protection. It intervenes when the tamper-resistance of the observer is broken. In the design of our payment system we follow this paradigm. Therefore, single-term coins are used, the form of which is the same as that proposed by Brands [1, 2]. The electronic wallet of the user can be either a workstation connected to the Internet or a small hand-held computer that is able to support an infra-red link with a Point-of-Sale (POS) terminal. The electronic wallet is equipped with a smartcard reader. The observer is represented by a smartcard issued by the bank to the user.

The main contribution of our solution consists of the speedup of the withdrawal protocol. The idea is that the withdrawal of coins is split in three separate transactions between the payer and the bank. These transactions are referred to as *get_pseudonym*, *withdraw_big_coin* and *exchange_big_coin*. The frequencies of execution of these transactions are different. Thus, the *get_pseudonym* transaction, which is time consuming, is executed seldom. The other two transactions are executed often but they are efficient. A *get_pseudonym* transaction is intended to provide the user with a set of pseudonyms. A pseudonym is validated by the bank, using the restrictive blind signature scheme introduced by Brands in [1, 2]. The pseudonym contains a certain invariant structure with respect to the blinding operations carried out by the user. The *withdraw_big_coin* transaction provides the user with a "big" coin, which is linked to a certain pseudonym of the user. In return his account (kept with the bank) is debited with the corresponding value of this coin. The user generates the big coin as a random number unknown to the bank, which signs it in connection with the associated pseudonym, using a blind version of the Schnorr signature issuing protocol. A such coin cannot be involved directly in a payment transaction. In order to make it spendable, the user has to exchange it at the bank, in the *exchange_big_coin* transaction. To this end, the user forwards to the bank a validated pseudonym together with a big coin linked to it. The bank assesses the validity of the pseudonym, verifies the authenticity of the big coin, and checks for it in a database containing all the big coins that were already exchanged. If the big coin to be exchanged is "fresh", the bank records it in the database, and pursues the *exchange_big_coin* transaction. At the end of this protocol, the user gets a number of coins which can be involved in subsequent payment transactions. Their cumulated values equate the value of the big coin from which they are obtained. The bank cannot trace a coin used in a certain payment transaction to a specific user. This holds even if the bank is considered unlimited powerful and colludes with shops, unless the user misuses a coin by involving it in more than one payment transaction. But, a set of coins obtained from the same big coin and embedding the same pseudonym are all linkable. If someone can link a pseudonym to a certain user, then the whole chain of payments involving coins related to that pseudonym can be traced back to the user. Therefore, the privacy of the user is somehow restricted. This drawback can be limited by "rotating" coins derived from different pseudonyms in a set of consecutive payment transactions. A similar construction, with a different implementation was proposed by Yacobi in [12].

The remainder of the paper is organized as follows. Section 2 introduces the notations and the main cryptographic assumptions. Afterwards, the shared signature scheme, used in the payment transaction, is formally described. In Section 3 the setup of the payment system is outlined. The last section details the protocols of the main transactions: *get_pseudonym*, *withdraw_big_coin*, *exchange_big_coin*, *payment* and *deposit*. Finally, some concluding remarks are presented.

2 Cryptographic Preliminaries

In this section, firstly the notations are introduced. Afterwards, the shared signature scheme, which is the basis of the payment transaction, is formalized.

2.1 Notations and Cryptographic Assumptions

For any prime p, the set of positive integers smaller than p is denoted $\mathbb{Z}_p^*$. Let q be a prime such that q divides $p-1$. The length of p is at least 512 bits, and that of q is at least 160 bits. We denote by G_q the unique subgroup of $\mathbb{Z}_p^*$ of order q, for which polynomial-time algorithms are known to determine equality of elements, to test membership, to compute inverses, to multiply and to randomly select elements. In expressions involving elements in G_q, we do not mention explicitly the reduction modulo p. This subgroup can be generated as follows:

1. choose at random $\gamma \in_{\mathcal{R}} \mathbb{Z}_p^*$;
2. compute g as $g \leftarrow \gamma^{(p-1)/q} \bmod p$;
3. if $g = 1$ then go to Step 1. Otherwise, compute G_q as $\{1, g, g^2, \ldots, g^{q-1}\}$.

The element g is named a *generator* of the subgroup G_q. For every $h \in G_q$, the unique index $x \in \mathbb{Z}_q$ such that $g^x = h \bmod p$ is denoted by $\log_g h$. The security of the electronic cash proposed in this paper relies on the discrete logarithm assumption for subgroups of prime order.

Assumption 2.1 *Finding the unique index* $\log_g h \in \mathbb{Z}_q$ *of* $h \in G_q$ *with respect to* $g \in G_q \backslash \{1\}$ *is the* discrete logarithm problem in subgroups of prime order. *An algorithm is said to solve this problem if, for inputs* $g \neq 1, h$ *generated uniformly at random, it outputs* $\log_g h$ *with at least non-negligible probability of success. The* discrete logarithm assumption for subgroups of prime order *states that there is no probabilistic polynomial-time algorithm that solves the aforesaid problem.*

During the generation of digital signatures, we use a hash function $\mathcal{H}(\cdot)$, mapping arbitrary long inputs to an output of fixed length $|\mathcal{H}|$, such that the following assumption is fulfilled:

Assumption 2.2 *The hash function* $\mathcal{H}(\cdot)$ *is selected such that it is collision-resistant and it has a pseudorandom behavior.*

2.2 The Shared Signature Scheme

The shared signature scheme represents the basis for the payment transaction. We consider the following two groups:

– $(G_q, \cdot, 1)$, where the operation of the group is defined by $x \cdot y = xy \bmod p$, and $1 \in \mathbb{Z}_p^*$ is the neutral element of the group;
– $(\mathbb{Z}_q \times \mathbb{Z}_q, \circ, e)$, where the operation of the group is defined by $(x, y) \circ (u, v) = (x+u \bmod q, y+v \bmod q)$, and the neutral element of the group is $e = (0, 0) \in \mathbb{Z}_q \times \mathbb{Z}_q$.

We also define the homomorphism f, mapping $\mathbb{Z}_q \times \mathbb{Z}_q$ to G_q such that $f(x, y) = g_1^x g_2^y$, where g_1, g_2 are different generators of G_q.

The shared signature scheme is formalized for the case when two signers, $\mathcal{T}_i$ and $\mathcal{C}_i$, join their secret keys s_t and s_c to generate the common secret key sk

and to issue the signature $\Sigma_{sk}(m)$ on a message m. A restriction is that $\mathcal{T}_i$ can communicate only to $\mathcal{C}_i$, who is the only one communicating to the verifier $\mathcal{V}$. An authority (within the payment system this authority is the bank) instantiates the homomorphism f by choosing (p, q, g_1, g_2). It also forwards to $\mathcal{T}_i$ a secret key $s_t \in \mathbb{Z}_q^*$ and to $\mathcal{C}_i$ the image p_t of $(s_t, 0)$ under f: $p_t = f(s_t, 0) = g_1^{s_t}$. $\mathcal{C}_i$ chooses at random $s_c \in \mathbb{Z}_q^*$ and computes the image p_c of $(s_c, 0)$ under f: $p_c = f(s_c, 0) = g_1^{s_c}$, which is stored by $\mathcal{C}_i$. In order to issue $\Sigma_{sk}(m)$ the verifier $\mathcal{V}$ and the two (cooperating) signers $\mathcal{T}_i$ and $\mathcal{C}_i$ follow the protocol:

1. $\mathcal{T}_i$ makes a proof of knowledge of the secret key s_t to $\mathcal{C}_i$, using the Schnorr identification scheme [11] with respect to $p_t = g_1^{s_t}$. To this end, $\mathcal{T}_i$ chooses at random $\xi_0 \in \mathbb{Z}_q^*$ and forwards the initial witness $B \leftarrow g_1^{\xi_0}$ to $\mathcal{C}_i$.
2. $\mathcal{C}_i$ generates at random $t, \xi_1, \xi_2 \in \mathbb{Z}_q^*$ and computes:

$$\begin{aligned} pk_1 &\leftarrow (p_t p_c g_2)^t = g_1^{ts_U} g_2^t, \text{where: } s_U = s_t + s_c \bmod q, \\ pk_2 &\leftarrow B^t g_1^{\xi_1} g_2^{\xi_2} = g_1^{\xi_1 + t\xi_0} g_2^{\xi_2}. \end{aligned}$$

   The above relations can be rewritten as follows:

$$\begin{aligned} pk_1 &= [f(s_t,0) \cdot f(s_c,1)]^t = f^t(s_t,0) \cdot f^t(s_c,1) = f(ts_t,0) \cdot f(ts_c,t) = \\ &= f((ts_t,0) \circ (ts_c,t)) = f(sk_{11} \circ sk_{12}) = f(ts_U,t) = f(sk_1), \\ pk_2 &= f^t(\xi_0,0) \cdot f(\xi_1,\xi_2) = f((t\xi_0,0) \circ (\xi_1,\xi_2)) = \\ &= f(sk_{21} \circ sk_{22}) = f(\xi_1 + t\xi_0, \xi_2) = f(sk_2). \end{aligned}$$

   Thus, the public key of the shared signature scheme is $pk = (pk_1, pk_2) \in G_q^2$ and the corresponding secret key is $sk = (sk_1, sk_2) \in (\mathbb{Z}_q \times \mathbb{Z}_q)^2$, where:

$$\begin{aligned} sk_1 &= sk_{11} \circ sk_{12} = (ts_t, 0) \circ (ts_c, t) = (ts_U, t), \\ sk_2 &= sk_{21} \circ sk_{22} = (t\xi_0, 0) \circ (\xi_1, \xi_2) = (\xi_1 + t\xi_0, \xi_2). \end{aligned}$$

   Therefore, in the above relations we can see the pairs

$$\begin{aligned} \{sk_{11} = (ts_t, 0), sk_{21} = (t\xi_0, 0)\}, \\ \{sk_{12} = (ts_c, t), sk_{22} = (\xi_1, \xi_2)\}. \end{aligned}$$

   as the contributions of $\mathcal{T}_i$ and $\mathcal{C}_i$ respectively, to the secret key sk. $\mathcal{C}_i$ sends pk (in a certified form) to $\mathcal{V}$.
3. $\mathcal{V}$ takes a random challenge α in $\mathbb{Z}_q$ and sends it to $\mathcal{C}_i$.
4. $\mathcal{C}_i$ computes the message to be signed $m \leftarrow \mathcal{H}(pk, \alpha)$ and sends it as a challenge to $\mathcal{T}_i$.
5. $\mathcal{T}_i$ computes the response $\rho_0 \leftarrow \xi_0 - ms_t \bmod q$ and sends it to $\mathcal{C}_i$. This ends the proof of knowledge of s_t provided by $\mathcal{T}_i$.
6. $\mathcal{C}_i$ accepts the proof if and only if $g_1^{\rho_0} p_t^m = B$. Using the response ρ_0 from $\mathcal{T}_i$'s proof, $\mathcal{C}_i$ can compute the signature on m as $\Sigma_{sk}(m) \leftarrow (\rho_1, \rho_2) \in \mathbb{Z}_q \times \mathbb{Z}_q$ where:

$$\begin{aligned} \rho_1 &\leftarrow \xi_1 - t(ms_c - \rho_0) = (\xi_1 + t\xi_0) - mts_U \bmod q, \\ \rho_2 &\leftarrow \xi_2 - mt \bmod q. \end{aligned}$$

Thus, the shared signature on m is

$$\begin{aligned}\Sigma_{sk}(m) &\leftarrow (\rho_1, \rho_2) = ((\xi_1 + t\xi_0) - mts_U, \xi_2 - mt) = \\ &= (ts_U, t)^{-m} \circ (\xi_1 + t\xi_0, \xi_2) = sk_1^{-m} \circ sk_2 .\end{aligned}$$

$\mathcal{C}_i$ sends $\Sigma_{sk}(m)$ to $\mathcal{V}$.

7. $\mathcal{V}$ computes $m \leftarrow \mathcal{H}(pk, \alpha)$ and accepts the signature $\Sigma_{sk}(m)$ if and only if $f(\Sigma_{sk}(m)) = {pk_1}^{-m} \cdot pk_2$.

3 Setup of the System

The participants in the system are the bank $\mathcal{B}$ and its clients, either users $\mathcal{U}_i$, playing the role of a payer, or shops $\mathcal{S}_j$, playing the role of a payee. In order to simplify the system, the case when users and shops are clients of different banks is not considered. During the transactions, a user $\mathcal{U}_i$ is represented by a combination of a personal workstation $\mathcal{C}_i$ and a tamper-resistant device $\mathcal{T}_i$. The workstation is controlled by the user. The tamper-resistant device is represented by a smartcard, which works as an observer for the bank.

3.1 System Parameters Generation

The bank generates the system parameters as follows:

– a hash function $\mathcal{H}(\cdot)$, mapping arbitrary long inputs to an output of fixed length $|\mathcal{H}|$, such that the Assumption 2.2 is fulfilled;
– the primes p and q such that $q|p-1$, $|q| > |\mathcal{H}|$, and the Assumptions 2.1 is satisfied;
– a generator-pair $(g_1, g_2) \in G_q^2$, $g_1 \neq g_2$, where G_q is the unique subgroup of order q of $\mathbb{Z}_p^*$;
– a generator $g \in G_q$ such that $g_1, g_2 \neq g$.

For security reasons, the relative logarithm of g_1 and g_2 must be unknown to users and shops.

The bank generates at random two pairs of keys. The pair (S, P), with the secret key $S \in_{\mathcal{R}} \mathbb{Z}_q^*$ and the corresponding public key $P = g^S$, is involved by the bank in two different signature issuing protocols:

– the restrictive blind signature issuing protocol, denoted $rb\sigma_S(\cdot)$. This protocol is used to validate the pseudonyms π for the user.
– a signature issuing protocol, denoted $\sigma_S(\cdot)$. This protocol is used by the bank to authenticate with respect to a user and to sign the coins C, generated during the exchange of a big coin.

The other pair (S_1, P_1), with $S_1 \in_{\mathcal{R}} \mathbb{Z}_q^*$ and $P_1 = g^{S_1}$, is used by the bank for an (ordinary) blind signature issuing protocol, denoted $b\sigma_S(\cdot)$. This protocol is used to authenticate the big coins BC withdrawn by the user.

The bank broadcasts through a variety of media the set of system parameters $(p, q, (g_1, g_2), g, \mathcal{H}(\cdot))$ together with the public keys P and P_1. Moreover, all the participants in the system can do arithmetics in the Abelian groups $(G_q, \cdot, 1)$ and $(\mathbb{Z}_q \times \mathbb{Z}_q, \circ, e)$. The homomorphism $f(x, y) = g_1^x g_2^y$, mapping $\mathbb{Z}_q \times \mathbb{Z}_q$ to G_q, is implicitly chosen according to the subset (p, q, g_1, g_2) of the system parameters.

The bank also manages three databases:

- the accounts database, dealing with information related to the users;
- the exchanged coins database, which records all the big coins that were already changed; and
- the transcripts database, which keeps the transcripts of the payment transactions.

3.2 Registration

When a person becomes a user $\mathcal{U}_i$ of the payment system he opens an account with $\mathcal{B}$, which generates an appropriate entry in the account database for $\mathcal{U}_i$. This entry records the account balance and the user's public keys $p_U = p_t p_c$ and p_c, where $p_t = g_1^{s_t}$, $p_c = g_1^{s_c}$. The public key p_U can be seen as $p_U = g_1^{s_t + s_c} = g_1^{s_U}$, where $s_U = s_t + s_c \bmod q$ is the joint secret key of the user. The bank generates at random the share $s_t \in \mathbb{Z}_q^*$ of s_U and issues to $\mathcal{U}_i$ a smartcard $\mathcal{T}_i$, which stores s_t. The bank also computes $p_t = g_1^{s_t}$, which is stored in the non-volatile memory of $\mathcal{C}_i$. The other share of s_U, denoted s_c, is generated at random in $\mathbb{Z}_q^*$ by $\mathcal{C}_i$, which also computes $p_c = g_1^{s_c}$ and forwards this item to the bank. $\mathcal{C}_i$ makes a proof of knowledge of $\log_{g_1} p_c$, and if the bank accepts this proof then the public keys $p_U = p_t p_c$ and p_c are uniquely linked to $\mathcal{U}_i$. Related to p_U, the bank computes other two items $\pi_0 = p_U g_2$ and $z_0 = \pi_0^S$, which are stored by $\mathcal{C}_i$ for subsequent use. The bank also stores (π_0, z_0) with the user's entry in the accounts database. The pair (π_0, z_0) can be seen as the invariant information related to the user.

The shop that subscribes to the payment system does not need a tamper-resistant device, but only has to be a client of $\mathcal{B}$.

4 Description of the Main Transactions

In this section we describe the main transactions of the proposed coin-based electronic cash system. We concentrate on the *get_pseudonym*, *withdraw_big_coin*, *exchange_big_coin*, *payment* and *deposit* transactions.

4.1 Validation of Pseudonyms

A *pseudonym* of the user $\mathcal{U}_i$ is a pair secret key/public key $(\mu = (\mu_1, \mu_2), \pi) \in \mathbb{Z}_q^2 \times G_q$. It is derived from the pair $(\mu_0 = (s_U, 1), \pi_0 = g_1^{s_U} g_2) \in \mathbb{Z}_q^2 \times G_q$, which is related to the secret key of the user s_U. The user generates at random $t \in \mathbb{Z}_q^*$, and computes $\pi = \pi_0^t = g_1^{t s_U} g_2^t$. The secret key corresponding to π can be seen as $\mu = (t s_U, t) \in \mathbb{Z}_q^2$. A *validated pseudonym* VP_i consists of the pair

$(\pi, rb\sigma_S(\pi))$, where π is restrictively blind signed by the bank $rb\sigma_S(\pi)$, such that the corresponding secret key μ still encodes the secret key s_U. In oder to provide the user with validated pseudonyms, the user and the bank repetitively execute the *get_pseudonym* protocol. Thus, the former obtains a number of validated pseudonyms VP_i, depending on the contract between the user and the bank. The *get_pseudonym* protocol is based on the restrictive blind signature scheme proposed by Brands [1, 2]. The protocol is shown in Figure 1, where the user $\mathcal{U}_i$ is represented only by $\mathcal{C}_i$. The smartcard $\mathcal{T}_i$ does not participate in this protocol.

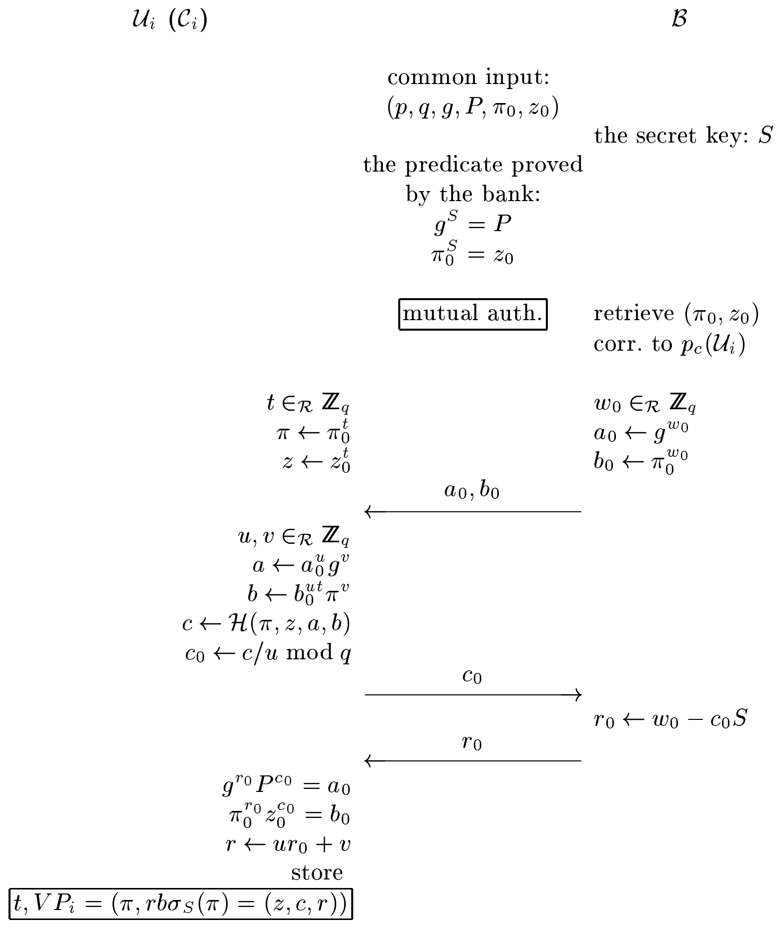
$\mathcal{U}_i$ $(\mathcal{C}_i)$
$\mathcal{B}$
common input: (p, q, g, P, π_0, z_0)
the secret key: S
the predicate proved by the bank: $g^S = P$, $\pi_0^S = z_0$
mutual auth.
retrieve (π_0, z_0) corr. to $p_c(\mathcal{U}_i)$
$t \in_{\mathcal{R}} \mathbb{Z}_q$, $\pi \leftarrow \pi_0^t$, $z \leftarrow z_0^t$
$w_0 \in_{\mathcal{R}} \mathbb{Z}_q$, $a_0 \leftarrow g^{w_0}$, $b_0 \leftarrow \pi_0^{w_0}$
a_0, b_0
$u, v \in_{\mathcal{R}} \mathbb{Z}_q$, $a \leftarrow a_0^u g^v$, $b \leftarrow b_0^{ut} \pi^v$, $c \leftarrow \mathcal{H}(\pi, z, a, b)$, $c_0 \leftarrow c/u \bmod q$
c_0
$r_0 \leftarrow w_0 - c_0 S$
r_0
$g^{r_0} P^{c_0} = a_0$, $\pi_0^{r_0} z_0^{c_0} = b_0$, $r \leftarrow ur_0 + v$
store
$t, VP_i = (\pi, rb\sigma_S(\pi) = (z, c, r))$

Fig. 1. The *get_pseudonym* transaction

In order to get a signature on the blinded version $\pi = \pi_0^t$, the user computes $z = \pi^S$ from z_0 as $z = z_0^t = (\pi_0^S)^t = (\pi_0^t)^S = \pi^S$. While the user carries out the protocol with input (g, P, π_0, z_0), he diverts the protocol with input (g, P, π, z).

Instead of generating a random challenge for the bank, the user computes the challenge as the output of the hash function $c \leftarrow \mathcal{H}(\pi, z, a, b)$. Previous to be sent to the bank, the challenge c is blinded to the value c_0, through the blinding parameter u. Finally, the user obtains the signature $rb\sigma_S(\pi) = (z, c, r)$ on π, where r is the response r_0 of the bank blinded through the parameters u, v (which are random choices of the user). The pseudonym π is considered valid if and only if $c = \mathcal{H}(\pi, z, g^r P^c, \pi^r z^c)$. The validated pseudonyms $VP_i = (\pi, rb\sigma_S(\pi) = (z, c, r))$ and the random parameter t are stored in the non-volatile memory of $\mathcal{C}_i$.

Before starting a session of repeated executions of the *get_pseudonym* protocol, the user and the bank mutually authenticate to one another. The user authenticates with respect to p_c and the bank authenticates with regard to P. For this purpose, both the user and the bank rely on the Schnorr signature scheme [11]. In this protocol, shown in Figure 2, the user is represented only by $\mathcal{C}_i$.

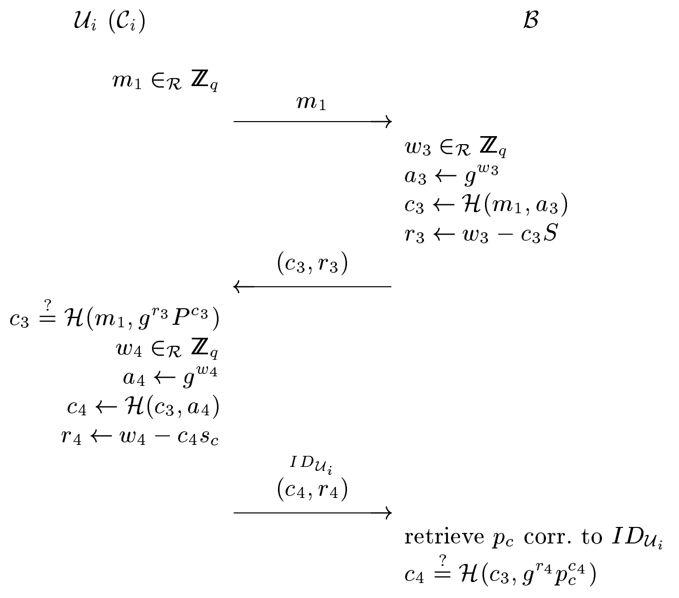
$\mathcal{U}_i$ $(\mathcal{C}_i)$
$\mathcal{B}$
$m_1 \in_{\mathcal{R}} \mathbb{Z}_q$
m_1
$w_3 \in_{\mathcal{R}} \mathbb{Z}_q$
$a_3 \leftarrow g^{w_3}$
$c_3 \leftarrow \mathcal{H}(m_1, a_3)$
$r_3 \leftarrow w_3 - c_3 S$
(c_3, r_3)
$c_3 \overset{?}{=} \mathcal{H}(m_1, g^{r_3} P^{c_3})$
$w_4 \in_{\mathcal{R}} \mathbb{Z}_q$
$a_4 \leftarrow g^{w_4}$
$c_4 \leftarrow \mathcal{H}(c_3, a_4)$
$r_4 \leftarrow w_4 - c_4 s_c$
$ID_{\mathcal{U}_i}$
(c_4, r_4)
retrieve p_c corr. to $ID_{\mathcal{U}_i}$
$c_4 \overset{?}{=} \mathcal{H}(c_3, g^{r_4} p_c^{c_4})$

Fig. 2. The mutual authentication protocol.

4.2 Withdrawal of Big Coins

A *big coin* BC consists of the tuple $(\beta, \pi, b\sigma_{S_1}(\beta, \pi))$, where β is randomly generated in $\mathbb{Z}_q$ by the user and π is a pseudonym of the user. In a separate session or continuing the session of issuing pseudonyms, the user and the bank repetitively execute the *withdraw_big_coin* protocol. In this protocol, which is presented in Figure 3, the bank signs in a blind way the items (β, π), issuing the signature $b\sigma_{S_1}(\beta, \pi)$. To this end, a blind version of the Schnorr signature scheme [11] is used. During the protocol, the user $\mathcal{U}_i$ is represented only by $\mathcal{C}_i$. The smartcard $\mathcal{T}_i$ does not participate in this protocol. The big coin $BC = (\beta, \pi, b\sigma_{S_1}(\beta, \pi) = (c_1, r_1))$ is considered authentic if $c_1 = \mathcal{H}(\beta, \pi, g^{r_1} P_1^{c_1})$. The items $(\beta, b\sigma_{S_1}(\beta, \pi))$ are stored in the non-volatile memory of $\mathcal{C}_i$. The current big coin is added to a list linked to the pseudonym π.

$\mathcal{U}_i$ $(\mathcal{C}_i)$		$\mathcal{B}$
	common input: (p, q, g, P_1)	
		the secret key: S_1
	$\boxed{\text{mutual auth.}}$	link to account$(\mathcal{U}_i)$
retrieve a pseudonym π		
$\beta \in_{\mathcal{R}} \mathbb{Z}_q$		$w_{01} \in_{\mathcal{R}} \mathbb{Z}_q$
		$a_{01} \leftarrow g^{w_{01}}$
	$\xleftarrow{a_{01}}$	
$u_1, v_1 \in_{\mathcal{R}} \mathbb{Z}_q$		
$a_1 \leftarrow a_{01} g^{v_1} P_1^{u_1}$		
$c_1 \leftarrow \mathcal{H}(\beta, \pi, a_1)$		
$c_{01} \leftarrow c_1 - u_1 \bmod q$		
	$\xrightarrow{c_{01}}$	
		$r_{01} \leftarrow w_{01} - c_{01} S_1$
	$\xleftarrow{r_{01}}$	
$g^{r_{01}} P_1^{c_{01}} \overset{?}{=} a_{01}$		
$r_1 \leftarrow r_{01} + v_1$		
store		
$\boxed{\beta, b\sigma_{S_1}(\beta, \pi) = (c_1, r_1)}$		
link the big coin to π		

Fig. 3. The *withdraw_big_coin* transaction

The actual number of big coins BC withdrawn is determined by the amount which the user agrees to debit from his account. If the withdrawal of big coins takes place in a separate session than that of getting validated pseudonyms, then the user and the bank mutually authenticate to one another. The user authenticates with respect to his public key p_c, such that the bank can deduct the equivalent value of the withdrawn coins from the appropriate account.

4.3 Exchange of a Big Coin

In a different session from those of getting validated pseudonyms or/and big coins, the user exchanges an authenticated big coin BC (obtained in connection with a pseudonym VP_i) for a number of coins C that can be involved in payment transactions. The user sends the big coin to be exchanged BC $(\beta, b\sigma_{S_1}(\beta, \pi))$ together with the pseudonym $VP_i = (\pi, rb\sigma_S(\pi) = (z, c, r))$ that is associated with BC. Firstly, the bank verifies in the corresponding database that $\beta(BC)$ is not already exchanged. Afterwards, the bank is authenticated by the user. In this way, the latter is convinced that he asks an exchange from the authorized bank and not from a bogus terminal. The bank verifies the validity of the pseudonym VP_i and checks the authenticity of the big coin BC, in relation with this pseudonym. If all these verifications hold true, the user is authenticated by the bank. To this end, the user signs β and a challenge from the bank with respect to the pseudonym π, in the framework of the Okamoto signature scheme [9]. If the signature is correct, the bank records the exchanged big coin (in fact $\beta(BC)$) in the database of already exchanged coins. The bank issues a number of spendable coins C such that their total values equate the value of the big coin forwarded. The protocol of the *exchange_big_coin* transaction is presented in Figure 4.

A *certified key pair*, *ckp*, is a triple consisting of a secret key $sk = (sk_1, sk_2) \in (\mathbb{Z}_q \times \mathbb{Z}_q)^2$, a public key $pk = (pk_1, pk_2) \in G_q \times G_q$, and a public-key certificate of the bank on the public key $\sigma_S(pk)$. Each electronic coin C that can be spent in the system is represented only by a pair consisting of the public key pk and the certificate of the bank on the public key $\sigma_S(pk)$, i.e., $C = (pk, \sigma_S(pk) = (c_2, r_2))$. The Schnorr signature scheme is used by the bank to issue the certificate of the coin. A coin is considered authentic if and only if $c_2 = \mathcal{H}(pk, g^{r_2} P^{c_2})$. We note that each coin C is linked to a certain pseudonym (μ, π), since the pseudonym is assigned to the pair (sk_1, pk_1) $(sk_1 \leftarrow \mu = (ts_U, t)$, $pk_1 \leftarrow \pi = g_1^{ts_U} g_2^t)$. The bank can see the public key pk of the coin to be certified. Still the bank cannot link this coin to a certain user, since the user is unknown during the protocol. The certificate allows the shop to verify the authenticity of a coin and its subsequent acceptability at the deposit stage.

4.4 Payment and Deposit

A payment using a coin C is a joint signature executed by $\mathcal{C}_i$ and $\mathcal{T}_i$, with respect to the coin's secret key sk. The coin C accepted by $\mathcal{C}_i$ on behalf of $\mathcal{U}_i$ during the withdrawal protocol can be spent with the shop $\mathcal{S}_j$ if the shop plays the verifier's role in the steps 3 to 7 in the shared signature scheme (see Section 2.2).

We denote by *pay_spec* the specification of a payment. This parameter includes the amount, the date and the identity of the shop involved in the payment transaction.

During deposit, $\mathcal{S}_j$ shows a transcript of a payment

$$(pk, c_2, r_2), (pay\_spec, \alpha, \rho_1, \rho_2)$$

$$
\begin{array}{rcccl}
\mathcal{T}_i & & \mathcal{C}_i & & \mathcal{B} \\
 & & \text{retrieve } BC\text{: } \beta, (c_1, r_1), & & \\
 & & VP_i(\beta) = (\pi, (z, c, r)), t & & \\
 & & m' \in_{\mathcal{R}} \mathbb{Z}_q & \xrightarrow{m', \beta, (c_1, r_1), VP_i(\beta)} & \text{Is } \beta(BC) \text{ fresh?} \\
 & & & & w' \in_{\mathcal{R}} \mathbb{Z}_q \\
 & & & & c' \leftarrow \mathcal{H}(m', g^{w'}) \\
 & & & & r' \leftarrow w' - c'S \\
 & & & \xleftarrow{(c', r')} & \\
 & & c' \overset{?}{=} \mathcal{H}(m', g^{r'} P^{c'}) & & VP_i \text{ valid?} \\
w_0 \in_{\mathcal{R}} \mathbb{Z}_q & & & & BC \text{ authentic?} \\
B_0 \leftarrow g_1^{w_0} & & w_1, w_2 \in_{\mathcal{R}} \mathbb{Z}_q & & \\
 & \xrightarrow{B_0} & a'' \leftarrow B_0^t g_1^{w_1} g_2^{w_2} & \xrightarrow{a''} & \\
 & & & & \alpha' \in_{\mathcal{R}} \mathbb{Z}_q \\
 & \xleftarrow{c''} & c'' \leftarrow \mathcal{H}(\beta, \alpha') & \xleftarrow{\alpha'} & \\
r_0 \leftarrow w_0 - c'' s_t & & & & \\
 & \xrightarrow{r_0} & & & \\
 & & r_1'' \leftarrow w_1 - t(c'' s_c - r_0) & & \\
 & & r_2'' \leftarrow w_2 - c'' t & & \\
 & & & \xrightarrow{(r_1'', r_2'')} & \\
 & & & & c'' \leftarrow \mathcal{H}(\beta, \alpha') \\
 & & & & a'' \overset{?}{=} g_1^{r_1''} g_2^{r_2''} \pi^{c''} \\
 & & & \xleftarrow{\text{OK}} & \text{record } \beta(BC) \\
 & & \textbf{execute repetitively} & & \\
\xi_0 \in_{\mathcal{R}} \mathbb{Z}_q & & & & \\
B \leftarrow g_1^{\xi_0} & & \xi_1, \xi_2 \in_{\mathcal{R}} \mathbb{Z}_q & & \\
 & \xrightarrow{B} & & & VP_i = (\pi, (z, c, r)) \\
 & & pk_2 \leftarrow B^t g_1^{\xi_1} g_2^{\xi_2} & & pk_1 \leftarrow \pi = (g_1^{t s_U} g_2^t) \\
 & & & \xrightarrow{pk_2} & \\
 & & & & pk \leftarrow (pk_1, pk_2) \\
 & & & & w_2 \in_{\mathcal{R}} \mathbb{Z}_q \\
 & & & & a_2 \leftarrow g^{w_2} \\
 & & & & c_2 \leftarrow \mathcal{H}(pk, a_2) \\
 & & c_2 \overset{?}{=} \mathcal{H}(pk, g^{r_2} P^{c_2}) & \xleftarrow{(c_2, r_2)} & r_2 \leftarrow w_2 - c_2 S \\
 & & \text{store coin:} & & \\
 & & \boxed{C = (pk, \sigma_S(pk) = (c_2, r_2))} & &
\end{array}
$$

Fig. 4. The *exchange_big_coin* transaction

to the bank. After the bank performs the same verifications that $\mathcal{S}_j$ has carried out in the payment to check the coin's authenticity, the coin is checked against double-deposit by $\mathcal{S}_j$ and against double-spending by the user. On one hand, if α is the same in two transcripts, $\mathcal{S}_j$ attempted to double-deposit and the bank refuses to refund the shop. On the other hand, if a coin is used in more than one payment transcript

$$(pk, c_2, r_2), (pay\_spec, \alpha, \rho_1, \rho_2),$$
$$(pk, c_2, r_2), (pay\_spec', \alpha', \rho_1', \rho_2').$$

the bank can derive the secret key of the dishonest user as

$$s_U = (\rho_1 - \rho_1')/(\rho_2 - \rho_2') \bmod q$$

and correspondingly user's real identity $p_U = g^{s_U}$.

5 Concluding Remarks

We have outlined the design of a coin-based electronic payment system, which offers full untraceability of the coins with respect to users, but all the coins related to a pseudonym of the user are linkable. This restricts somehow the privacy of the user, but the efficiency of the system is increased, by the speedup of the withdrawal transaction. In our solution, the withdrawal is executed in three separate transactions, with different frequencies of execution. The time consuming transaction is executed seldom, while the transactions which are often executed are efficient by the point of view of execution time.

References

1. S. Brands. Untraceable off-line cash in wallet with observers. In *Advances in Cryptology—CRYPTO '93*, volume 773 of *Lecture Notes in Computer Science*, pages 302–318, Berlin, 1993. Springer-Verlag.
2. S. Brands. An efficient off-line electronic cash system based on the representation problem. Report CS-R9323, Centrum voor Wiskunde en Informatica, March 1993.
3. D. Chaum, B. den Boer, E. van Heyst, S. Mjølsnes, and A. Steenbeek. Efficient offline electronic checks. In *Advances in Cryptology—EUROCRYPT '89*, volume 434 of *Lecture Notes in Computer Science*, pages 294–301, Berlin, 1990. Springer-Verlag.
4. D. Chaum, A. Fiat, and M. Naor. Untraceable electronic cash. In *Advances in Cryptology—CRYPTO '88*, volume 403 of *Lecture Notes in Computer Science*, pages 319–327, Berlin, 1990.. Springer-Verlag.
5. D. Chaum and T. P. Pedersen. Wallet databases with observers. In *Advances in Cryptology—CRYPTO '92*, volume 740 of *Lecture Notes in Computer Science*, pages 89–105, Berlin, 1993. Springer-Verlag.
6. N. T. Ferguson. Single term off-line coins. In *Advances in Cryptology—EUROCRYPT '93*, volume 765 of *Lecture Notes in Computer Science*, pages 318–328, Berlin, 1993. Springer-Verlag.

7. M. Franklin and M. Yung. Towards provably secure efficient electronic cash. Technical report CUCS-018-92, Columbia University, Dept. of Computer Science, April 1992.
8. M. Franklin and M. Yung. Secure and efficient off-line digital money. In *Automata, Languages and Programming, ICALP 93*, volume 700 of *Lecture Notes in Computer Science*, pages 265–276, Berlin, 1993. Springer-Verlag.
9. T. Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. In *Advances in Cryptology—CRYPTO '92*, volume 740 of *Lecture Notes in Computer Science*, pages 31–53, Berlin, 1993. Springer-Verlag.
10. T. Okamoto and K. Ohta. Universal electronic cash. In *Advances in Cryptology—CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 324–337, Berlin, 1992. Springer-Verlag.
11. C. P. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, 1991.
12. Y. Yacobi. Efficient electronic money. In *Advances in Cryptology—ASIACRYPT '94*, volume 917 of *Lecture Notes in Computer Science*, pages 153–164, Berlin, 1994. Springer-Verlag.