An Efficient Divisible Electronic Cash Scheme

Tatsuaki Okamoto

NTT Laboratories
1-2356 Take, Yokosuka-shi, 238-03 Japan
Email: okamoto@sucaba.isl.ntt.jp

Abstract. Recently, several "divisible" untraceable off-line electronic cash schemes have been presented [8, 11, 19, 20]. This paper presents the first practical "divisible" untraceable[1] off-line cash scheme that is "single-term"[2] in which every procedure can be executed in the order of $\log \mathcal{N}$, where $\mathcal{N}$ is the precision of divisibility, i.e., $\mathcal{N}$ = (the total coin value)/(minimum divisible unit value). Therefore, our "divisible" off-line cash scheme is more efficient and practical than the previous schemes. For example, when $\mathcal{N} = 2^{17}$ (e.g., the total value is about \$ 1000, and the minimum divisible unit is 1 cent), our scheme requires only about 1 Kbyte of data be transfered from a customer to a shop for one payment and about 20 modular exponentiations for one payment, while all previous divisible cash schemes require more than several Kbytes of transfered data and more than 200 modular exponentiations for one payment.
In addition, we prove the security of the proposed cash scheme under some cryptographic assumptions. Our scheme is the first "practical divisible" untraceable off-line cash scheme whose cryptographic security assumptions are theoretically clarified.

1 Introduction

Recently, much research has been performed in the area of off-line electronic currency [2, 5, 8, 9, 11, 12, 13, 16, 18, 19, 20, 25]. Protocols have been proposed enabling consumers to withdraw "electronic coins" from a bank, and later spend these coins at a shop in an "off-line" manner. Here, off-line refers to the property that communication with a bank or authorized center is unnecessary during the payment protocol. In addition, electronic coins should be anonymous, i.e., "untraceable".

A "divisible" coin worth some amount of money, say \$x, is a coin that can be spent many times as long as the sum total of all its the transactions does not exceed \$x. This property, divisibility, is very useful and convenient for a

[1] Note that coins divided from the same coin can be linked each other in the proposed scheme, although they are anonymous, i.e., "untraceable" from the customer's identity. In other words, the unlinkability among divided coins is not satisfied, although the untraceability is satisfied.

[2] In the first generation of the practical off-line cash schemes [5, 16, 18, 19, 20], the cut-and-choose method is used, in which cash consists of many terms of the same form (e.g., 40 terms). A "single-term" cash scheme [2, 11, 12] means a practical cash scheme in which the cut-and-choose method is not used and cash consists of a single term. The basic idea of "single-term" is from [13], but the technique to realize the "single-term" property is specific to each scheme [2, 11, 12].

D. Coppersmith (Ed.): Advances in Cryptology - CRYPTO '95, LNCS 963, pp. 438-451, 1995.

customer. If a coin is not divisible, the customer must withdraw a coin whenever he spends it, or withdraw many coins of various values and store them in his electronic wallet (e.g., smart card). Since real cash does not satisfy this property, we must use many various bills and coins in daily life. On the other hand, prepaid cards[3] satisfy this property, and this is the major merit of such cards over real cash. Therefore, in practice, the divisibility is a very important requirement for electronic cash systems.

So far, several "divisible" untraceable off-line electronic cash schemes have been presented [8, 11, 19, 20]. The scheme in [8] is far from practical, since their scheme utilizes a non-interactive zero-knowledge proof for a general NP predicate. Pailles' divisible coin construction [20] is also inefficient. The size of data transfered from a customer to a shop during payment is linear in $\mathcal{N}$ where $\mathcal{N}$ is the divisibility precision, i.e., $\mathcal{N}$ = (the total coin value)/(minimum divisible unit value). A system in which a coin worth \$5367 consists of 5367 \$1 coins is a rather unwieldy and inefficient divisible cash system. [19] has also shortcomings in efficiency: their scheme utilizes a cut-and-choose method, a paid/deposited coin consists of many terms (e.g., 40 terms), and hence, the resulting complexities (the transfered data size and computation amount for a payment) can be quite large.

[11] partially solved the efficiency problems by constructing the first "single-term" divisible cash scheme. The cut-and-choose method is not used, so the transfered data size of [11] is less than that of [19]. However, the major shortcoming of [11] is that the required computation amount for a payment is of the order of the divisibility precision, $\mathcal{N}$. Hence, the amount of computation required for a payment in [11] is almost as large as that of [19]. Another shortcoming of [11] is that the size of the divisible coin depends on the selection of routes in the binary tree. If the selection of routes is almost optimal, the transfered data can be much smaller than that of [19]. However, if no such selection exists (such cases often occur), the transfered data size becomes almost as large as that of [19].

This paper presents a divisible untraceable off-line electronic cash scheme which solves these shortcomings, and is much more efficient than all previous schemes. Our scheme is the first practical "single-term divisible" cash scheme in which every procedure can be executed in the order of $\log \mathcal{N}$, and every transfered data sizes are of the order of $\log \mathcal{N}$. Therefore, the amount of the required computation and communication for a payment is, with our scheme, much less than those of the previous schemes.

For example, when $\mathcal{N} = 2^{17}$ (e.g., the total value is about \$ 1000, and the minimum divisible unit is 1 cent), our scheme requires only about 1 Kbyte of data be transfered from a customer to a shop for one payment and about 20 modular exponentiations (for a customer and a shop respectively) for one payment. All previous practical divisible cash schemes require more than several Kbytes of transfered data and more than 200 modular exponentiations for one payment. [4]

---

[3] Prepaid cards such as telephone cards are a kind of electronic cash, but their security heavily depends on physical tricks. Here, electronic cash means a cash system whose security depends only on mathematical techniques. In this sense, electronic cash can be considered to be an ideal version of prepaid cards and so is suitable for smart cards.

[4] Note that the evaluation here is based on the degree of security of factoring a 512 bit composite (and a 512 bit modulus discrete logarithm). For example, if the modulus size is 1024 bits, then the data sizes of the cash schemes should be twice. On the other hand, the number of the modular exponentiations can only depend on an arbitrary

The withdrawal procedure of our scheme is very efficient; it requires around 0.1 Kbyte of data be exchanged between a customer to a bank for one withdrawal and just one modular exponentiation (for a customer and a bank respectively) for one withdrawal, regardless of the total value and the divisibility precision. Moreover, the amount of data (electronic license and coin) stored in an electronic wallet (e.g., a smart card) is around 0.2 Kbytes. All previous practical divisible cash schemes require more than several Kbytes be stored in a wallet.

In addition, we prove that the proposed cash scheme satisfies four security requirements (no forging, no tracing, no overspending and no swindling) under some assumptions. Our scheme is the first "practical divisible" untraceable cash scheme whose cryptographic security assumptions, which are relatively primitive, are theoretically clarified. Even from a practical viewpoint, such a security proof is very important especially for a complicated cryptographic protocol, which consists of many primitive protocols. Although it is unclear whether these four security requirements are sufficient, our security proof guarantees that if there exists an attack on our scheme, then it should reside outside these four security requirements, unless our security assumptions are broken.

Note that the unlinkability among coins divided from the same coin cannot be satisfied in the proposed scheme as well as the previous practical divisible untraceable off-line cash schemes, although these divided coins are anonymous, i.e., "untraceable". In addition, since the procedure for interrupting the linking chain among the withdrawn coins (i.e., the opening protocol) is less efficient than the other procedures of our scheme, the procedure cannot be executed so often.

2 Number Theoretic Conventions

Since our scheme is constructed using some number theoretic techniques developed by [19], this paper follows the notations and propositions of the number theoretic techniques in [19]. However, Lemma 1 and Lemma 2 are new in this paper (A similar technique is used in [3]). They constitute a new technique to prevent a customer from double-spending a coin (or a node of a tree).

Lemma 1. *Let $N = PQ$ be the Williams integer, and t be an integer which is greater than 1. Then, for any $x \in QR_N$, and for any $e \in Z_{2^t}$, there exists a unique solution y such that $y^{2^t} = 2^{2^e} x \bmod N$ and $y \in QR_N$.*

Lemma 2. *Let $N = PQ$ be the Williams integer, and t be an integer which is greater than 1 and $t = O(|N|)$. Then, there exits a deterministic poly-time (i.e., $O(|N|^3)$) algorithm to factor N, given N, t, $x \in QR_N$, $e_1 \in Z_{2^t}$, $e_2 \in Z_{2^t}$, $(e_1 \neq e_2)$, y_1, and y_2 such that*

$$y_i^{2^t} \equiv 2^{2^{e_i}} x \bmod N \quad \textit{and}$$

$$y_i \in QR_N \ (i = 1, 2).$$

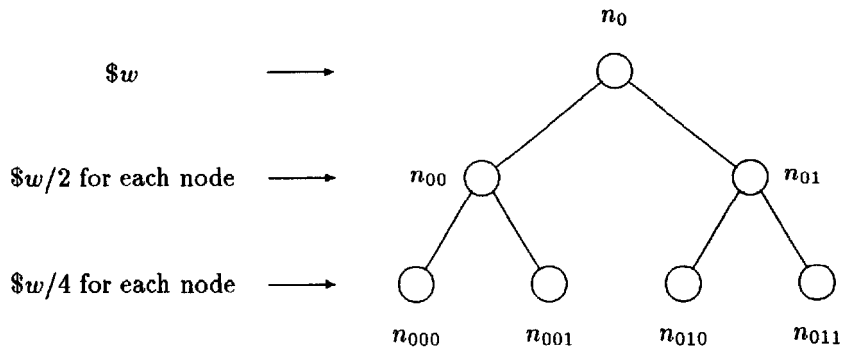
---

security parameter and $\mathcal{N}$ practically, although, theoretically, it also depends on the modulus size. In our evaluation, this security parameter, the probability that a dishonest Customer is accepted in the payment protocol, is supposed to be at most $1/2^{40}$.

3 Binary Tree Approach

We will adopt the binary tree approach as do all previous divisible cash schemes [8, 11, 19, 20]. Each coin of worth $w = 2^l$ is associated with a tree of $(1 + l)$ levels and w leaves.

Each node of the tree represents a certain denomination. The root node, n_0, is assigned a monetary value of w, and the value of all other nodes, $n_{j_1 \cdots j_l}$, is found by halving the value of the node's parent, $n_{j_1 \cdots j_{l-1}}$ ($j_1 = 0$, $j_i \in \{0, 1\}$ for $i = 2, \ldots, l$).



With this tree, we will show that for a single coin of worth w, it will be possible for a consumer to engage in several transactions, such that the sum total of the amounts of each transaction is less than or equal to w.

Divisibility can be implemented under the following two rules:

1. (**Route node rule:**) When a node is used, all descendant nodes and all ancestor nodes of this node cannot be used.
2. (**Same node rule:**) No node can be used more than once.

Preserving both rules implies that the set of past transactions involving the coin is legitimate and vice versa. Spending more than $\$w$, the value of a coin, will result in violation of at least one of these rules.

Moreover, in our concrete cash scheme, which will be shown in the following sections, two values are used for each node in the tree (Γ value and Λ value); Γ values are used to realize the route node rule, and Λ values to realize the same node rule. $\Gamma_{j_1 \cdots j_l}$ and $\Lambda_{j_1 \cdots j_l}$ denote Γ value and Λ value for node $n_{j_1 \cdots j_l}$ respectively. In addition, Ω values are introduced to calculate Γ values.

4 Bit Commitment Schemes

A bit commitment scheme is used in the opening stage of the proposed scheme, in place of the cut-and-choose method of the previous schemes [19, 20]. That is, the commitment scheme plays an essential role in realizing the "single-term" property of the proposed cash scheme.

One type of commitment scheme was used in the gradually releasing protocol [7]. Our paper, however, uses another type of commitment scheme, since the commitment scheme in [7] is not appropriate in our scheme. The new commitment scheme is based on the discrete logarithm problem [5], while the scheme in [7] is based on the factoring problem. However, almost all techniques developed in [7] can be used with slight modification in our scheme.

4.1 Bit Commitments

Assume that B sets up the commitment scheme and U commits to a number. Finally U proves to B that a value is correctly generated without revealing committed information, by using some protocols to be described later.

To set up the commitment scheme, B generates prime $\mathcal{P}$ satisfying $\mathcal{P} - 1 = 2 \cdot Prime$ ($Prime$ is a prime number), G and g whose orders in the multiplicative group $Z_{\mathcal{P}}^*$ are $Prime$. B sends $\mathcal{P}$, G and g. U checks whether $Prime = (\mathcal{P}-1)/2$ is a prime by a probabilistic primality (or composite) test, and whether the orders of G and g are $Prime$ by checking that they are not 1 and $G^{Prime} \equiv 1 \pmod{\mathcal{P}}$ and $g^{Prime} \equiv 1 \pmod{\mathcal{P}}$.

U can commit to any integer $s \in Z_{Prime}$ by choosing R uniformly at random in Z_{Prime} and computing the commitment

$$BC_g(R, s) = G^R g^s \bmod \mathcal{P}.$$

This is called a base-g commitment. A commitment is opened by revealing R and s.

4.2 Protocols of Checking the Contents of Bit Commitments

This subsection introduces some useful protocols in which U can prove to B in a zero-knowledge manner that a committed value is in an interval, and that two committed values are equivalent.

Let the interval be $I = [a, b]$ $(= \{x | a \leq x \leq b\})$, $e = b - a$, and $I \pm e = [a - e, b + e]$.

Protocol: CHECK COMMITMENT

Common input: x and $(\mathcal{P}, G, g, I)$.

What to prove: U knows (R, s) such that $x = BC_g(R, s)$ and $s \in I \pm e$.

Execute the following k times:

1. U chooses t_1 uniformly in $[0, e]$, and sets $t_2 = t_1 - e$. U sends to B the unordered pair of commitments $T_1 = BC_g(S_1, t_1)$, $T_2 = BC_g(S_2, t_2)$.
2. B selects a bit $\beta \in \{0, 1\}$ and sends it to U.
3. U sends to B one of the following:
   (a) if β is 0, opening of both T_1 and T_2
   (b) if β is 1, opening of $x \cdot T_i \bmod N$ $(i \in \{1, 2\})$, such that $s + t_i \in I$.
4. B checks the correctness of U's messages.

Protocol: COMPARE COMMITMENTS

Common input: x, x' and $(\mathcal{P}, G, g, I)$.

What to prove: U knows (R, R', s) such that $x = BC_g(R, s)$, $x' = BC_g(R', s)$ and $s \in I \pm e$.

Execute the following k times:

[5] The underlying technique has been well known (e.g., [6, 21]).

1. U chooses t_1 uniformly in $[0, e]$, and sets $t_2 = t_1 - e$. U sends to B the unordered pair of commitments (T_1, T_1'), (T_2, T_2'), where each component of the pair is ordered and $(T_i, T_i') = (BC_g(S_i, t_i), BC_h(S_i', t_i))$.
2. B selects a bit $\beta \in \{0, 1\}$ and sends it to U.
3. U sends to B one of the following:
   (a) if β is 0, opening of both (T_1, T_1') and (T_2, T_2')
   (b) if β is 1, opening of $x \cdot T_i \bmod N$ and $x' \cdot T_i' \bmod N$ $(i \in \{1, 2\})$, such that $s + t_i \in I$.
4. B checks the correctness of U's messages.

Protocol: CHECK MOD-MULT
Common input: x, y, z, n and $(\mathcal{P}, G, g, I = [n, 2n])$. $(|(\mathcal{P}-1)/2| \geq 2|n|+6)$
What to prove: U knows $(R, R', R'', s, t, \alpha)$ such that $x = BC_g(R, s)$, $y = BC_g(R', t)$, $z = BC_g(R'', \alpha)$, $\alpha \equiv st \pmod{n}$, and $s, t, \alpha \in [0, 3n] (= I \pm n)$.

1. U uses the CHECK COMMITMENT protocol with $I = [n, 2n]$ to prove that U knows how to open x to reveal a value in $[0, 3n]$ $(= I \pm n)$.
2. U sends $v = BC_x(R''', t) = BC_g(R''' + tR, st)$, and uses the COMPARE COMMITMENTS protocol with I to prove that U knows how to open $y = BC_g(R, t)$ and $v = BC_x(R''', t)$.
3. U sends $u = BC_g(R'''', d)$, where d is defined by $st = \alpha + dn$, $\alpha \equiv st \pmod{n}$, and $s, t, \alpha \in I$.
4. U uses the CHECK COMMITMENT protocol with $[n-1, 4n-1]$ to prove that U knows how to open u to reveal a value in $[-2n-1, 7n-1]$ $(= [n-1, 4n-1] \pm 3n)$. U uses the CHECK COMMITMENT protocol with I to prove that U knows how to open z to reveal a value in $I \pm n$.
5. U opens (as a base-g commitment) the product $zu^n v^{-1} \bmod \mathcal{P}$ to reveal a 0 (i.e., reveals R^* such that $BC_g(R^*, 0) = zu^n v^{-1} \bmod \mathcal{P}$).

5 Efficient Divisible Cash Scheme

This section outlines the various protocols in our divisible electronic cash scheme.

As in [18, 19], the electronic cash in our proposed scheme consists of an electronic license and electronic coins.

The electronic license is issued by the bank to a customer during an "opening protocol". This protocol is done once per customer, typically when a customer opens an account. If, however, a customer prefers to change the license at some later time, or desires several licenses, this protocol is run again for each additional license. As mentioned in [18], the frequency of license changes should be determined after considering the trade-offs between the degree of unlinkability and efficiency desired[6].

5.1 The Opening Protocol

As a result of the one-time opening protocol, customer U obtains an *electronic license* $(N, L_1 = (N+a_1)^{1/K} \bmod n_1, L_2 = (N+a_2)^{1/K} \bmod n_2)$. This basically grants U permission to use the electronic cash of bank B. Here, (n_1, K) and

[6] Even if a customer does not change the license at all, the customer's privacy (no traceability) is mathematically preserved as shown in Theorem 4. Then, anonymous purchase histories with the same license are linkable.

(n_2, K) are B's RSA public keys and (a_1, a_2) is also B's public key. They are common to many (or all) customers. (a_i, n_i, K) $(i = 1, 2)$ are used to check the validity of U's license, (N, L_1, L_2), in the payment and deposit protocols. N and (L_1, L_2) are kept secret from B in this opening protocol, to keep the untraceability.

Roughly, in this protocol, U secretly generates N, which is the composite of two primes P and Q, and gives $x = g^P \bmod \mathcal{P}$ and $y = g^Q \bmod \mathcal{P}$ to B as U's identity, where $\mathcal{P}$ and g are B's public key, which can be common for many customers. Then, U asks B to sign N in a blind manner (through the RSA blind signature), after U proves B that N is honestly generated (in the relation with x and y) in a zero-knowledge manner. So, finally U gets B's RSA signature, (L_1, L_2), for N, while B has no information on N and (L_1, L_2).

The opening protocol is as follows:

1. Bank B chooses the parameters of the bit commitment scheme, $\mathcal{P}$, G and g, and sends them to Customer U. B also sends RSA public-keys, (n_1, K) and (n_2, K), and public key (a_1, a_2), as the public keys of B's blind signatures. Here, K is the form of $2^J + 1$ (J: integer). For simplicity of explanation here, for $i = 1, 2$, we assume that $n_i = p_i q_i$ (p_i, q_i: prime) and $(p_i - 1)/2$ and $(q_i - 1)/2$ are primes. (When n_i is a more general form, a slight modification is required for the value of K such that, for example, K must be prime.) We also assume $|n_1| = |n_2|$ and $|a_i| < |n_i|/2$. W.l.o.g, we assume $n_1 < n_2$.
2. Customer U checks whether $Prime = (\mathcal{P} - 1)/2$ is a prime, and whether the orders of G and g are $Prime$. U also checks whether $|(\mathcal{P} - 1)/2| \geq 2|n_i| + 6$. U generates two primes P and Q, and random numbers, $r_i \in Z_{n_i}$ $(i = 1, 2)$. Here, $|P| < K$ and $|Q| < K$, and $(1/4){n_1}^{1/2} < P, Q < (1/2){n_1}^{1/2}$. U calculates $x = g^P \bmod \mathcal{P}$, $y = g^Q \bmod \mathcal{P}$, and $s_i = (N + a_i){r_i}^K \bmod n_i$ $(i = 1, 2)$. U sends (x, y, s_1, s_2) to B with U's signature, as U's identity.
3. U and B execute the following protocol: (Informally, U proves to B in a zero-knowledge manner that (s_1, s_2, x, y) is honestly generated.)
   (a) U generates random numbers, R_N, $R_{r_i}^{(0)}$, $R_{r_i}^{(1)}$, $\cdots$, $R_{r_i}^{(J)}$, $R_{r_i}^{*} \in Z_{Prime}$, $(i = 1, 2)$.
   U calculates $\alpha_i^{(1)}$, $\alpha_i^{(2)}$, $\cdots$, $\alpha_i^{(J)}$ such that $\alpha_i^{(j)} \equiv {r_i}^{2^j} \pmod{n_i}$ $(i = 1, 2; j = 1, \cdots, J)$ $(\alpha_i^{(0)} = r_i)$ and $\alpha_i^{(j)} \in [n_i, 2n_i]$ $(i = 1, 2; j = 0, \cdots, J)$.
   U also calculates α_i^{*} such that $\alpha_i^{*} \equiv {r_i}^K \pmod{n_i}$ $(\equiv {r_i}^{2^J+1} \pmod{n_i})$ and $\alpha_i^{*} \in [n_i, 2n_i]$.
   U also sends B the following values: $z = BC_x(R_N, Q) = BC_g(R_N, N)$, $(N = PQ)$, $u_i^{(0)} = BC_g(R_{r_i}^{(0)}, \alpha_i^{(0)})$, ..., $u_i^{(J)} = BC_g(R_{r_i}^{(J)}, \alpha_i^{(J)})$, $u_i^{*} = BC_g(R_{r_i}^{*}, \alpha_i^{*})$.
   (b) U uses the CHECK COMMITMENT protocol with interval $[(1/4){n_1}^{1/2}, (1/2){n_1}^{1/2}]$ to prove that U knows how to open x to reveal a value in $[0, (3/4){n_1}^{1/2}]$. (Although x is a specific type of bit commitment: $x = BC_g(0, P)$, the CHECK COMMITMENT protocol can be used similarly.)
   (c) U uses the COMPARE COMMITMENTS protocol with $[(1/4){n_1}^{1/2}, (1/2){n_1}^{1/2}]$ to prove that U knows how to open $y = BC_g(0, Q)$ and $z = BC_x(R_N, Q)$ in $[0, (3/4){n_1}^{1/2}]$.
   (d) The following procedure is repeated for $j = 0, \cdots, J - 1$ and $i = 1, 2$:

U uses the MOD-MULT protocol with $[n_i, 2n_i]$ for $(u_i^{(j)}, u_i^{(j)}, u_i^{(j+1)}, n_i, \mathcal{P}, G, g)$ to prove that $\alpha_i^{(j+1)}$ is committed to by $u_i^{(j+1)}$. (Duplicated procedures between j-th step and $(j+1)$-th step such as CHECK COMMITMENT protocol for $u_i^{(j)}$ can be omitted.)

(e) U uses the MOD-MULT protocol with $[n_i, 2n_i]$ for $(u_i^{(0)}, u_i^{(J)}, u_i^*, n_i, \mathcal{P}, G, g)$ to prove that α_i^* is committed to by u_i^*.
U uses the MOD-MULT protocol with $[n_i, 2n_i]$ for $(z_i, u_i^*, BC_g(0, s_i), n_i, \mathcal{P}, G, g)$ to prove that $s_i = (N + a_i) r_i^K \bmod n_i$ $(i = 1, 2)$, where $z_i = z g^{a_i} \bmod \mathcal{P} = BC_g(R_N, N + a_i)$.

4. B gives $\delta_i = s_i^{1/K} \bmod n_i$ $(i = 1, 2)$ to U.
5. U obtains $L_i = (N + a_i)^{1/K} \bmod n_i$ by $\delta_i / r_i \bmod n_i$ $(i = 1, 2)$.

5.2 The Withdrawal Protocol

When the customer wants to withdraw $\$w$ from the account, an electronic coin of worth $\$w$ is then obtained by executing the withdrawal protocol with the bank.

The withdrawal protocol itself is very simple: Bank B just issues a blind signature to user U. Assume the consumer wishes to withdraw a divisible coin worth $w = 2^l$ dollars from his account at bank B. (That is, assume U sends B U's signed message to request the withdrawal. Here, we assume that the key for U's signature is independently generated from the other parameters except the size.) Also, B has a public key of the RSA signatures, (e_w, n_w), which corresponds to $w = 2^l$ dollars. The following steps occur:

1. U chooses a random value b, then forms and sends Z to B.

$$Z = r^{e_w} H(N \parallel b) \bmod n_w,$$

where $r \in Z_{n_w}$ is a random integer and H is a one-way hash function.

2. B gives $Z^{1/e_w} \bmod n_w$ to U and charges U's account $\$w$.
3. U can then extract the electronic coin $C = (H(N \parallel b))^{1/e_w} \bmod n_w$.

5.3 Payment

Assume that customer U spends $\$y$ $(\leq w)$ at shop V through the payment protocol. The payment protocol consists of two stages: coin authentication and denomination revelation. During the coin authentication phase, the shop verifies that the coin bears the bank's signature. During the second phase, the customer reveals information about a certain set of nodes in the coin's binary tree representation depending on the denomination being spent. We assume that f_Γ, f_Ω, f_A and h are truly random (or pseudo random) functions[7]. These stages are described in more detail as follows:

[7] Here we omit the explicit description of the domains and codomains of these functions, since they are naturally determined by the input and output variables.

Coin Authentication Customer U supplies shop V with (L_1, L_2), N, (C, b) and w. The shop checks that ${L_i}^K \equiv N + a_i \pmod{n_i}$ $(i = 1, 2)$, $C^{e_w} \equiv H(N \parallel b) \pmod{n_w}$, $(-1/N) = 1$, and $(2/N) = -1$.

If the number, k, of the nodes corresponding to U's payment to V is less than 10 ($k' = 10 - k > 0$)[8], V gives $z_1, \cdots, z_{2k'}$ to U, U calculates $[(< f_\Gamma(z_1) >_{QR})^{1/2} \bmod N]_1, \cdots, [(< f_\Gamma(z_{2k'}) >_{QR})^{1/2} \bmod N]_1$, and send them to V. V checks the validity.

Denomination Revelation Let $[y_{l+1} y_l \cdots y_1]$ $(y_i \in \{0, 1\}, i = 1, \ldots, l+1)$ be the binary representation of y. Here the length, $l+1$, of the binary string is fixed by the coin value ($w = 2^l$), and some most significant bits such as y_{l+1} can be 0. Then, if $y_{l+2-t} = 1$ $(t = 1, \ldots, l+1)$, U selects a node $n_{j_1 \cdots j_t}$ [9] among the nodes in the t-th level that do not violate the two binary tree rules (see Section 3). Here, U has memorized the nodes already spent. The average number of nodes to be spent per a payment is at most $(l+1)/2$.

We will show the payment protocol when U spends node $n_{0 j_1 j_2 \cdots j_t}$ to V. When several nodes are spent per a payment, the following protocol of each node must be executed simultaneously.

1. U computes $\Gamma_{j_1 \cdots j_t}$,

$$\Gamma_{j_1 \cdots j_t} = [(< (\Omega_{j_1 \cdots j_{t-1}})^{2^{t-1} j_t} (\Omega_{j_1 \cdots j_{t-2}})^{2^{t-2} j_{t-1}} \cdots (\Omega_{j_1})^{2 j_2} \times$$

$$f_\Gamma(C \parallel 0 \parallel N) >_{QR})^{1/2^t} \bmod N]_{-1},$$

   where $\Omega_{j_1 \cdots j_i} =< f_\Omega(C \parallel j_1 \parallel \cdots \parallel j_i \parallel N) >_1$ $(i = 1, \ldots, t-1)$.
2. V computes $\Omega_{j_1 \cdots j_i}$ when $j_{i+1} = 1$ $(i = 1, \ldots, t-1)$. Then V verifies the validity of $\Gamma_{j_1 \cdots j_t}$ such that

$$(\Gamma_{j_1 \cdots j_t} / N) = -1,$$

$$(\Gamma_{j_1 \cdots j_t})^{2^t} \equiv$$

$$d(\Omega_{j_1 \cdots j_{t-1}})^{2^{t-1} j_t} (\Omega_{j_1 \cdots j_{t-2}})^{2^{t-2} j_{t-1}} \cdots (\Omega_{j_1})^{2 j_2} f_\Gamma(C \parallel 0 \parallel N) \pmod{N},$$

   where $d \in \{\pm 1, \pm 2\}$. If they are valid, V selects a random value $e' \in \{0, 1\}^u$, and sends V's identity ID_V, time T, and e' to U, where $u = O(m)$, $m = |P|(= |Q|)$. Otherwise V halts this protocol. V computes $e = h(ID_V \parallel T \parallel e')$, where $e \in \{0, 1\}^u$.
3. U computes $e = h(ID_V \parallel T \parallel e')$. U also computes $\Lambda_{j_1 \cdots j_t}$ such that

$$(\Lambda_{j_1 \cdots j_t})^{2^{u+1}} \equiv 2^{2e} < f_\Lambda(C \parallel j_1 \parallel \cdots \parallel j_t \parallel N) >_{QR} \pmod{N}.$$

4. V verifies that

$$(\Lambda_{j_1 \cdots j_t})^{2^{u+1}} \equiv d' 2^{2e} f_\Lambda(C \parallel j_1 \parallel \cdots \parallel j_t \parallel N) \pmod{N},$$

   where $d' \in \{\pm 1, \pm 2\}$. If verification succeeds, V accepts U's messages as payment of the amount due.

[8] The value, 10, is theoretically $O(m)$, where $m = |P|(= |Q|)$. This is required for checking whether N is the Williams integer with high probability.

[9] If U selects a node randomly among the valid nodes, U can conceal some information about U's purchase history to V, since if nodes to be spent are selected by a published rule, V may get some information about the purchase history from the nodes.

Remarks
1. Notations $\Gamma_{j_1 \cdots j_t}$ and $\Lambda_{j_1 \cdots j_t}$ are differently used from those in [19]. ($\Gamma_{j_1 \cdots j_t}$ and $\Lambda_{j_1 \cdots j_t}$ in this paper correspond to $X_{i,j_1 \cdots j_t}$ and $Y_{i,j_1 \cdots j_t}$ in [19].)
2. Function h can be a collision intractable hash function in place of a truly random (or pseudo random) function.

5.4 Deposit

Deposit is as before; a transcript of payment is forwarded to the bank.

5.5 Detection of Overspending

Although, formally, the security including the detection of overspending is described in Section 6, in this subsection, we will describe the detection procedure of overspending.

Clearly, if U overspends a coin, U must violate one of the two rules of the binary tree approach (see Section 3).

First, we show that "Route node rule" of the binary tree approach is securely realized. Assume that nodes $n_{0 j_1 j_2 \cdots j_i}$ and $n_{0 j_1 j_2 \cdots j_i \cdots j_t}$ are used. (Clearly this assumption violates the the route node rule.) Then U sends $\Gamma_{j_1 \cdots j_i}$ and $\Gamma_{j_1 \cdots j_t}$ to shops, and these values are finally sent to B. B can firstly detect that the violation of the rule occurs, by checking the coin values (C, b) along with (L_1, L_2, N) and the consumed nodes, $n_{0 j_1 j_2 \cdots j_i}$ and $n_{0 j_1 j_2 \cdots j_i \cdots j_t}$, in B's data base. (To efficiently find the violation practically, B can use a short hashed values (e.g., 32 bytes) of the coin values as a search key in the data base.)

Then,

$$\Gamma_{j_1 \cdots j_i} = [(< (\Omega_{j_1 \cdots j_{i-1}})^{2^{i-1} j_i} \cdots (\Omega_{j_1})^{2 j_2} f_\Gamma(C \parallel 0 \parallel N) >_{QR})^{1/2^i} \bmod N]_{-1},$$

On the other hand, from $\Gamma_{j_1 \cdots j_t}$, B computes

$$(\Gamma_{j_1 \cdots j_t})^{2^{t-i}} \equiv ((\Omega_{j_1 \cdots j_{t-1}})^{2^{t-i-1} j_t} \cdots (\Omega_{j_1 \cdots j_i})^{2^0 j_{i+1}}) \times$$

$$[(< (\Omega_{j_1 \cdots j_{i-1}})^{2^{i-1} j_i} \cdots (\Omega_{j_1})^{2 j_2} f_\Gamma(C \parallel 0 \parallel N) >_{QR})^{1/2^i} \bmod N]_1. \pmod N,$$

since $(\Gamma_{j_1 \cdots j_t})^{2^{t-i}} \bmod N$ is the quadratic residue and $(\Omega_{j_1 \cdots j_i}/N) = 1$. Therefore, B can compute

$$[(< (\Omega_{j_1 \cdots j_{i-1}})^{2^{i-1} j_i} \cdots (\Omega_{j_1})^{2 j_2} f_\Gamma(C \parallel 0 \parallel N) >_{QR})^{1/2^i} \bmod N]_1.$$

Using this value and $\Gamma_{j_1 \cdots j_i}$, B can efficiently and deterministically factor N and obtains P and Q, from which B can trace U's identity, x and y. Here, (P, Q) is the witness of U's violating one of these rules.

Next, we show that "Same node rule" of the binary tree approach is also securely realized. Assume that a node $n_{0 j_1 j_2 \cdots j_t}$ is used twice at different time or place. Then U's challenge messages (say, e_1 and e_2) of the double spending should be different with overwhelming probability from the property of a random function, h). Then, clearly from Lemma 2, B can efficiently factor N and obtains P and Q, from which B can trace U's identity, x and y.

6 Security

In this section, we show that the proposed cash scheme satisfies the four security requirements under some assumptions.

First, we introduce the security requirements for our cash system. They are a modification of those of [13].

Definition 3. Let m be $|P| = |Q|$. The proposed cash system is secure if the following conditions are satisfied:

- **No forging:** For all integers $k > 0$, for any poly-time nonuniform algorithm *Adv* (e.g., dishonest customer), after *Adv*'s k times execution as a customer of the (resp., opening, withdrawal) protocol with bank B, the probability that *Adv* computes $k+1$ (resp., licenses, coins) that pass the coin authentication by a shop is negligible in m.
- **No tracing:** For any poly-time nonuniform algorithm *Adv* (e.g., dishonest bank), after *Adv*'s execution as a bank and shops of the opening, withdrawal and payment protocols with customers U_0 and U_1, and given a payment transcript by customer U_r ($r \in \{0,1\}$), the probability that *Adv* outputs r correctly is less than $1/2 + 1/m^c$ for all constant c and for all sufficiently large m.
- **No overspending:** Suppose that customer U withdraws a coin, C, worth w dollars through the valid opening and withdrawal protocols with bank B. For any possible value of w, if customer U spends more than w dollars by C through payment protocol with shops, then there exists a probabilistic poly-time algorithm, *DETECT* (e.g., bank), which, given all payment transcripts regarding C, can compute P such that $x = g^P \bmod \mathcal{P}$ with overwhelming probability in m, where x is U's identity authorized in the opening protocol.
- **No swindling:** For any possible value of w, for any poly-time nonuniform algorithm *Adv* (e.g., dishonest shop), given all transcripts of opening and withdrawal protocols and after *Adv*'s executions as a shop of the payment protocol with customers, in which the total payment value is w dollars, the probability that *Adv* can deposit more than w dollars at Bank B is negligible in m.

Remark: In *No overspending* condition, *DETECT* can not only trace U's identity x from the overspending payment transcripts, but also gives the evidence, P, that U cannot deny U's overspending, since any poly-time algorithm is hard to calculate P unless U overspends.

Next, the following assumptions are required to prove the security of the proposed cash scheme.

Assumptions:

- **(RSA signatures)**
  - RSA signatures used with one-way hash functions are existentially unforgeable against adaptive chosen message attacks[10].
  - Let (n_1, K) and (n_2, K) be two RSA public keys such that n_1 and n_2 are independently selected, and $|n_1| = |n_2|$. Let $a_1, a_2 \in \{0,1\}^{|n_1|/2}$ be independently selected and published. Let $L_1^K \equiv N + a_1 \pmod{n_1}$ and $L_2^K \equiv N + a_2 \pmod{n_2}$. Then, the signature (L_1, L_2) for message N is existentially unforgeable against adaptive chosen message attacks.

[10] For the definition of "existentially unforgeable" etc., see [15].

- There is no efficient algorithm, given an RSA public key, (n, K), to output P and Z with $P < n^{1/2}$, $P \equiv Z^K \pmod{n}$, and $K > |P| > 1$.

– (**Factoring and Diffie-Hellman**) Let $\mathcal{P}$, P_0, Q_0, P_1 and Q_1 be primes, $N_0 = P_0Q_0$, $N_1 = P_1Q_1$, and $m = |P_0| = |Q_0| = |P_1| = |Q_1| < |\mathcal{P}| < dm$ for a constant d. Let $(\mathcal{P}-1)/2$ be a prime, and the order of g in the multiplicative group $Z^*_{\mathcal{P}}$ is $(\mathcal{P}-1)/2$. Then, for any probabilistic poly-time (non-uniform) machine M, given $\mathcal{P}$, g, $(x_0 = g^{P_0} \bmod \mathcal{P}, y_0 = g^{Q_0} \bmod \mathcal{P})$, $(x_1 = g^{P_1} \bmod \mathcal{P}, y_1 = g^{Q_1} \bmod \mathcal{P})$, and (N_r, N_{1-r}) $(r \in_R \{0,1\})$, M can compute r with probability less than $1/2 + 1/m^c$ for all constant c and for all sufficiently large m.

– (**Random functions**) Commonly available functions, f_Γ, f_Λ, f_Ω and h are truly random or pseudo-random[11].

Remarks

1. The first and second assumptions seem to be reasonable in practice, but the third assumption may be controversial, since a truly random function cannot be realized in the real world (as more than an exponential size of memory is required), and a commonly available pseudo-random function requires a tamper-free device. However, by using this assumption, the theoretical security requirements for our scheme can be clarified. We believe that a truly random or pseudo-random function f can be replaced by a practical one-way hash function family without sacrificing the security of our scheme, in practice.
2. The first assumption implies that factoring an RSA modulus n is hard, since if it is solved, the first assumption does not hold. The second assumption implies that both factoring N_i and the discrete logarithm $g^{P_i} \bmod \mathcal{P}$ are hard, since if at least one of them is solved, the second assumption does not hold.

Theorem 4. *The proposed cash scheme is secure under the above-mentioned assumptions.*

7 Efficiency

For this section, let us assume that $|P| = |Q| = 256$ bits, $|N| = 512$ bits, $|n_i| = 514$ bits $(i = 1, 2)$, $(\ (1/4)n_1^{1/2} < P, Q < (1/2)n_1^{1/2})$, $|Prime| \geq 1030$ bits, $|b| = 64$ bits, and $|n_w| = 512$ bits. Then, $K = 2^8 + 1$ (i.e., $J = 8$). We also assume the binary tree has 18 levels, i.e., $\mathcal{N} = 2^{17}$, where $\mathcal{N} =$ (the total coin value)/(minimum divisible unit value). For example, the total value is about \$ 1000, and the minimum divisible unit is 1 cent. Here, we also assume that efficient random hash functions are used, which are usually much faster than modular exponentiations.

Then, customer U uses 576 bits (72 bytes) of data for the electronic coin (C, b) worth \$1000 and U's proper data (electronic license, L_1, L_2, P, Q) is 1536 bits (192 bytes). Thus the total amount of data (264 bytes) is small enough to be stored on typical smart cards.

In the withdrawal protocol, customer U and bank B send 512 bit (64 bytes) messages respectively, and U and B just compute one exponentiation $\bmod n_w$ respectively.

[11] For the definitions of truly random and pseudo-random functions, see [14].

In the payment protocol (denomination revelation), 9 nodes are used on average for each payment when the tree has 18 levels [12]. For each node, two 512 bit values ($\Gamma_{j_1 \cdots j_t}$ and $\Lambda_{j_1 \cdots j_t}$) are transfered to the shop. Hence, a total of 1152 bytes is transfered to a shop in a payment on average. The computation required from a customer U for a payment is, in total, 18 (2×9) times the computation of the 2^u-th (or 2^{u+1}-th) root mod$N$, using the factors of N. Each root computation mod$N$ is almost comparable to exponentiation mod$N$. The computation required from the shop is almost the same, 18 exponentiations mod$N$.

The most time consuming part of our cash system is the opening protocol, although the protocol is still practical, as the amount of computation and communication is $O((\log |P|)|P|^4)$. But, this protocol can be executed much less frequently than the other procedures such as withdrawal and payment protocols (see Subsection 5.1). When $J = 8$ and $k = 20$, around 4000 multi-exponentiations mod$\mathcal{P}$ are required (from U and B respectively) for the opening protocol. (A multi-exponentiation can be computed almost as efficiently as a exponentiation by the extended binary method [17].) The amount is fairly heavy, but all of U's computation can be pre-computed, and the opening protocol can be executed by U's terminal (Workstation or PC) instead of a smart card. (After the opening protocol, U can store the data (just 192 bytes) in a smart card.)

8 Conclusion

This paper has presented a practical "divisible" off-line electronic cash scheme that is more efficient than previous schemes. Our scheme is the first practical divisible cash scheme that is single-term and in which every procedure can be executed in the logarithmic order of the precision of divisibility.

In addition, we proved the security of the proposed cash scheme under some cryptographic assumptions. Our scheme is the first practical divisible coin scheme whose cryptographic security assumptions are theoretically clarified.

The remaining problems are:

- Improve the efficiency of the opening protocol.
- Realize the unlinkability among coins divided from the same coin.
- Prove the security under more primitive assumptions such as the hardness of factoring and discrete logarithm.
- Find requirements which are formally shown to be sufficient for the security of electronic cash schemes. (The four requirements shown in this paper are still ad hoc.)

Acknowledgments

I would like to thank Ivan Damgård, Torben Pedersen and Moti Yung for informing me some attacks on preliminary versions of this paper. I would also like to thank Tony Eng for many useful comments. This work was partially conducted while visiting AT&T Bell Laboratories, Murray Hill, NJ, USA. I wish to thank Andrew Odlyzko and Bell Laboratories for the hospitality.

[12] In the coin authentication phase of the payment protocol, the modular square root operations are needed with probability of 4/18, and the average number of the root operations is 5. So, the average amount of the computation and communication is comparable to 2.5 ($= 10 \times (10/18) \times (1/2)$) node operations. Here, we neglect the operations of this phase, to simplify the evaluation, since it is relatively small.

References

1. Blum, M., "Coin flipping by telephone", IEEE, COMPCON, pp.133-137 (1982).
2. Brands, S., "Untraceable Off-line Cash in Wallet with Observers", Proceedings of Crypto 93, pp.302–318 (1994).
3. Bleumer, G., Pfitzmann, B. and Waidner, M., "A Remark on a Signature Scheme Where Forgery can be Proved", Proceedings of Eurorypt 90, pp.441–445 (1991).
4. Chaum, D., "Security without Identification: Transaction Systems to Make Big Brother Obsolete," Comm. of the ACM, 28, 10, pp.1030–1044 (1985).
5. Chaum, D., Fiat, A., and Naor, M., "Untraceable Electronic Cash," Proceedings of Crypto 88, pp.319-327 (1990).
6. Chaum, D., van Heijst, E., and Pfitzmann, B., "Cryptographically Strong Undeniable Signatures, Unconditionally Secure for the Signer," Proceedings of Crypto 91, pp.470-484 (1992).
7. Damgård, I., "Practical and Provably Secure Release of a Secret and Exchange of Signatures," Proceedings of Eurocrypt 93 (1993).
8. D'amingo, S. and Di Crescenzo, G., "Methodology for Digital Money based on General Cryptographic Tools", to appear in the Proceedings of Eurocrypt 94.
9. De Santis, A. and Persiano, G., "Communication Efficient Zero-Knowledge Proofs of Knowledge (with Applications to Electronic Cash)" Proceedings of STACS 92, pp. 449-460 (1992).
10. Even, S., Goldreich, O. and Yacobi, Y., "Electronic Wallet", Proceedings of Crypto 83, pp.383-386 (1983).
11. Eng, T. and Okamoto, T. "Single-Term Divisible Coins," to appear in the Proceedings of Eurocrypt 94.
12. Ferguson, N., "Single Term Off-line Coins", Proceedings of Eurocrypt 93, pp.318–328 (1994).
13. Franklin, M. and Yung, M., "Secure and Efficient Off-Line Digital Money", Proceedings of ICALP 93, pp. 449-460 (1993).
14. Goldreich, O., Goldwasser, S., and Micali, S., "How to Construct Random Functions," Journal of ACM, Vol.33, No.4 (1986).
15. Goldwasser, S., Micali, S. and Rivest, R., "A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks," SIAM J. Comput., 17, 2, pp.281–308 (1988).
16. Hayes, B., "Anonymous One-Time Signatures and Flexible Untraceable Electronic Cash," Proceedings of Auscrypt 90, pp.294-305 (1990).
17. Knuth, D.E. *The Art of Computer Programming*, Vol.2, 2nd Ed. Addison-Wesley (1981).
18. Okamoto, T., and Ohta, K., "Disposable Zero-Knowledge Authentication and Their Applications to Untraceable Electronic Cash", Proceedings of Crypto 89, pp. 481-496 (1990).
19. Okamoto, T., and Ohta, K., "Universal Electronic Cash", Proceedings of Crypto 91, pp. 324-337 (1992).
20. Pailles, J.C., "New Protocols for Electronic Money", Proceedings of Auscrypt 92, pp. 263–274 (1993).
21. Pedersen, T. P., "Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing", Proceedings of Crypto 91, pp. 129–140 (1992).
22. Pfitzmann, B. and Waidner, M., "How to Break and Repair a "Provably Secure" Untraceable Payment System," Proceedings of Crypto 91 (1992).
23. Rabin, M.O., "Digitalized Signatures and Public-Key Functions as Intractable as Factorization," Tech. Rep., MIT/LCS/TR-212, MIT Lab. Comp. Sci., (1979).
24. Vaudenay, S., "One-Time Identification with Low Memory," Eurocodes 92 (1992).
25. Yacobi, Y., "Efficient electronic money", to appear in the Proceedings of Asiacrypt 94.