Untraceable Electronic Cash †

(Extended Abstract)

David Chaum[1] *Amos Fiat*[2] *Moni Naor*[3]

[1] Center for Mathematics and Computer Science
Kruislaan 413, 1098 SJ Amsterdam, The Netherlands

[2] Tel-Aviv University
Tel-Aviv, Israel

[3] IBM Almaden Research Center
650 Harry Road, San Jose, CA 95120

Introduction

The use of credit cards today is an act of faith on the part of all concerned. Each party is vulnerable to fraud by the others, and the cardholder in particular has no protection against surveillance.

Paper cash is considered to have a significant advantage over credit cards with respect to privacy, although the serial numbers on cash make it traceable in principle. Chaum has introduced unconditionally untraceable electronic money([C85] and [C88]). But what is to prevent anyone from making several copies of an electronic coin and using them at different shops? On-line clearing is one possible solution though a rather expensive one. Paper banknotes don't present this problem, since making exact copies of them is thought to be infeasible. Nor do credit cards, because their unique identity lets the bank take legal action to regain overdrawn balances, and the bank can add cards to a blacklist.

Generating an electronic cash should be difficult for anyone, unless it is done in cooperation with the bank. The RSA digital signature scheme can be used to realize untraceable electronic money as proposed in [C85 and C88]. This money might be of the form $(x, f(x)^{1/3} \pmod n)$ where n is some composite whose factorization is known only to the bank and f is a suitable one-way function. The protocol for issuing and spending such money can be summarized as follows:

1. Alice chooses a random x and r, and supplies the bank with $B = r^3 f(x) \pmod n)$.

† Work done while the second and third authors were at the University of California at Berkeley. The work of the second author was supported by a Weizmann Postdoctoral Fellowship and by NSF Grants DCR 84-11954 and DCR 85-13926. The work of the third author was supported by NSF Grants DCR 85-13926 and CCR 88-13632.

S. Goldwasser (Ed.): Advances in Cryptology - CRYPTO '88, LNCS 403, pp. 319-327, 1990.

2. The bank returns the third root of B modulo n: $r \cdot f(x)^{1/3} \pmod{n}$ and withdraws one dollar from her account.
3. Alice extracts $C = f(x)^{1/3} \bmod n$ from B.
4. To pay Bob one dollar, Alice gives him the pair $(x, f(x)^{1/3} \pmod{n})$.
5. Bob immediately calls the bank, verifying that this electronic coin has not already been deposited.

Everyone can easily verify that the coin has the right structure and has been signed by the bank, yet the bank cannot link this specific coin to Alice's account.

Among other advantages, the new approach presented here removes the requirement that the shopkeeper must contact the bank during every transaction. If Alice uses a coin only once, her privacy is protected unconditionally. But if Alice reuses a coin, the bank can trace it to her account and can *prove* that she has used it twice.

Our work is motivated by that on minimum disclosure ([C86], [BC86a], [BC86b] and [BCC]) and on zero-knowledge ([GMR], [GMW86a] and [GMW86b]). Our scheme protects Alice's privacy unconditionally as is possible with the former, rather than computationally as in the latter. Using these very general results – which seem to be infeasible in practice – the security of the protocols presented here could be reduced to, say factoring (or any onw-way permutation if Alice's privacy is only computationally secure). Instead, We use the cut-and-choose methodology (first introduced in [R77]) directly, yielding quite practical constructions.

The next section presents our basic scheme, which guarantees untraceability, yet allows the bank to trace a "repeat spender". We then show how to modify the protocol so that the bank can supply incontestable proof that Alice has reused her money. Finally, we give a more efficient variant and briefly discuss further work.

1. Untraceable Coins

The bank initially publishes an RSA modulus n whose factorization is kept secret and for which $\phi(n)$ has no small odd factors. The bank also sets some security parameter k.

Let f and g be two-argument collision-free functions; that is, for any particular such function, it is infeasible to find two inputs that map to the same point. We require that f be "similar to a random oracle". For unconditional untraceability we also require g to have the property that fixing the first argument gives a one-to-one (or c to 1) map from the second argument *onto* the range.

Alice has a bank account numbered u and the bank keeps a counter v associated with it. Let $\oplus$ denote bitwise exclusive or and $\|$ denote concatenation.

To get an electronic coin, Alice conducts the following protocol with the bank:

1. Alice chooses a_i, c_i, d_i and r_i, $1 \le i \le k$, independently and uniformly at random from the residues $\pmod n$.
2. Alice forms and sends to the bank k *blinded candidates* (called B for mnemonic purposes)
$$B_i = {r_i}^3 \cdot f(x_i, y_i) \bmod n \qquad \text{for} \qquad 1 \le i \le k,$$
where
$$x_i = g(a_i, c_i) \qquad y_i = g(a_i \oplus (u \| (v+i)), d_i).$$
3. The bank chooses a random subset of $k/2$ blinded candidate indices $R = \{i_j\}$, $1 \le i_j \le k$ for $1 \le j \le k/2$ and transmits it to Alice.
4. Alice displays the r_i, a_i, c_i and d_i values for all i in R, and the bank checks them. Note that $u \| (v+i)$ is known to the bank. To simplify notation we will assume that $R = \{k/2+1, k/2+2, \ldots, k\}$.
5. The bank gives Alice
$$\prod_{i \notin R} B_i^{1/3} = \prod_{1 \le i \le k/2} B_i^{1/3} \bmod n$$
and charges her account one dollar. The bank also increments Alice's counter v by k.
6. Alice can then easily extract the electronic coin
$$C = \prod_{1 \le i \le k/2} f(x_i, y_i)^{1/3} \bmod n.$$
Alice reindexes the candidates in C to be lexicographic on their representation: $f(x_1, y_1) < f(x_2, y_2) < \cdots < f(x_{k/2}, y_{k/2})$. Alice also increments her copy of the counter v by k.

Note: For any fixed ϵ, if fewer than $(1-\epsilon)$ of the k blinded candidates B_i's have the proper form $(r^3 f(g(a_i, c_i), g(a_i \oplus (u \| (v+i)), d_i)))$, then Alice is caught with probability $1 - \exp(-c\epsilon k)$ for some constant c.

To pay Bob one dollar, Alice and Bob proceed as follows:

1. Alice sends C to Bob.
2. Bob chooses a random binary string $z_1, z_2, \ldots, z_{k/2}$.
3. Alice responds as follows, for all $1 \le i \le k/2$:
   a. If $z_i = 1$, then Alice sends Bob a_i, c_i and y_i.
   b. If $z_i = 0$, then Alice sends Bob x_i, $a_i \oplus (u \| (v+i))$ and d_i.
4. Bob verifies that C is of the proper form and that Alice's responses fit C.
5. Bob later sends C and Alice's responses to the bank, which verifies their correctness and credits his account.

The bank must store C, the binary string $z_1, \ldots, z_k$ and the values a_i (for $z_i = 1$) and $a_i \oplus (u \| v)$ (for $z_i = 0$).

If Alice uses the same coin C twice, then she has a high probability of being traced: with high probability, two different shopkeepers will send complementary binary values for at least one bit z_i for which B_i was of the proper form. The bank can easily search its records to ensure that C has not been used before. If Alice uses C twice, then, with high probability, the bank has both a_i and $a_i \oplus (u \| (v+i))$ with high probability. Thus, the bank can isolate u and trace the payment to Alice's account.

A possible problem with this scheme is a collusion between Alice and a second shopkeeper Charlie. After the transaction with Bob, Alice describes the transaction to Charlie, and both Bob and Charlie send the bank the same information; the bank knows that with very high probability one of them is lying, but has no way of telling which one, and cannot trace the coin to Alice's account

By fixing Bob's challenge to Alice, however, such a coalition can be kept from defrauding the bank. Every shopkeeper has a fixed query string, and every two strings have Hamming distance at least ck for some constant c. To prevent Alice from reusing the same coin at the same shop part of the challenge should still be random, or the shopkeeper should maintain his own list.

The scheme we describe above requires Alice to hold several coin denominations and use them to pay the exact amount. Section 3 presents a more efficient way to handle exact amounts.

2. Proving Multiple Spending

The scheme we describe above has the unfortunate property that the bank can frame Alice as a multiple spender. This means that these schemes cannot have any legal significance. To prevent a frame-up we assume that Alice has a digital signature scheme and a certified copy of her public key. Because we use digital signatures, Alice is protected against frame-up only computationally, not unconditionally. Yet, Alice's privacy remains unconditionally protected.

Rather than use the same account number u for all coins given to Alice, u will vary from coin to coin and from one blinded candidate to the next. We describe only the modifications to the basic scheme of section one.

Alice chooses two random integers z_i' and z_i'' for every i; u_i could then be chosen of the form "Alice's Account Number" $\|$ z' $\|$ z''. Along with the blinded candidates (the B_i values) Alice supplies the bank with a digital signature on

$$g(z_1', z_1'') \| g(z_2', z_2'') \| \cdots \| g(z_k', z_k'').$$

During the cut-and-choose, the bank verifies that each of the $k/2$ B_i's it examines generate an appropriate u_i. The bank has legal proof that Alice reused the coin whenever it can present the preimage of at least $k/2+1$ of the $g(z_i', z_i'')$.

Of course Alice has no hope if the bank can break the signature scheme she has chosen. Assuming the bank cannot forge her signature, then even if the bank can break g, its bijective property mentioned earlier ensures, with high probability, that she can prove g was broken by showing her (z_i', z_i'') for any broken $g(z_i', z_i'')$. This is a proof, since the assumption is that only the bank and not Alice can break g.

3. Untraceable Checks

The following scheme emulates the concept of guaranteed checks (similar to that of EuroChecks), but ensures untraceability. Alice requests a set of checks, whereby she can use each check for any single amount up to its limit and can later request a refund for the difference (limit minus actual sum). The bank will not know where the money was spent, nor the individual transaction amounts.

Alice can generate several checks in one interaction with the bank. The checks are similar to the basic version described in section one, but the first j factors are used to encode the purchase sum and the next $k-j$ factors are used to prevent Alice from using any check more than once.

The bank publishes two different RSA moduli, n and n', which are used for two different kinds of digital signature.

Alice's u can be used either as in section one or in section two. As before, let v be Alice's personal counter.

Alice sends the bank t pairs of major and minor candidates. For every major candidate Alice chooses b, c, d, and a at random; a major candidate M_i is of the form $f(x,y)$ where $x = g(a\|b,c)$ and $y = g(a \oplus (u\|(v+i)),d)$. Each minor candidate is of the form $g(b,e)$ where e is chosen at random. Alice generates several major candidates $M_1, M_2, \ldots, M_t$ and their related minor candidates $m_1, m_2, \ldots, m_t$.

Alice blinds the major and minor terms before submitting them to the bank. Blinded major candidates are of the form $B(M_i) = r^{3^k} \cdot M_i \bmod n$, where r is chosen at random; blinded minor candidates are of the form $B(m_i) = r^{3^k} \cdot m_i \bmod n'$. If the bank provides some $3^i th$ root of a blinded major(minor) term, $i \leq k$, then, as before, Alice can extract the appropriate root of the major(minor) term itself.

Alice sends the blinded M_i's and m_i's to the bank. Much as in section two, the bank performs a cut-and-choose operation, verifying that 1/2 of the pairs have the proper form. Then the bank performs a random permutation of the rest, grouping them into ordered sets of size k. Let one such set be denoted for simplicity

$B(M_1), B(M_2), \ldots, B(M_k)$. The bank extracts the following roots:

$$F_i = B(M_i)^{1/3^i} \pmod{n} \qquad \text{for} \quad 1 \le i \le k,$$
$$D_i = B(m_i)^{1/3^i} \pmod{n'} \qquad \text{for} \quad 1 \le i \le j.$$

The bank now returns the product of the k roots of blinded major candidates ($\prod_{i=1}^{k} F_i$); the appropriate roots of the j blinded minor candidates are returned individually. Alice extracts the check

$$C = \prod_{i=1}^{k} M_i^{1/3^i}$$

and $E_1, E_2, \ldots, E_j$, where $E_i = m_i^{1/3^i}$.

The bank now increments Alice's counter v by t, Alice does likewise to her local copy.

To make a purchase with such a check Alice encodes the purchase sum by regarding the first j of the M_i locations as denominations $1, 2, \ldots, 2^{j-1}$. If the ith denomination is a term in the purchase sum, then Alice reveals to the shopkeeper the appropriate y_i and the preimages of the x_i; if the ith denomination is not a term in the purchase sum, then Alice reveals x_i and y_i. Thus, later presenting E_i and the internal structure of the matching m_i term to the bank for a refund is safe exactly when the denomination is not spent.

Note: Given a root of the form $x^{1/3^i}$, it is trivial to compute roots of the form $x^{1/3^j}$ for $j \le i$. Thus, Alice could use the denomination 2^j, not use the denomination 2^i, $j < i$, and present the bank with the value

$$E_i^{i-j} = (m_i^{1/3^i})^{i-j} = m_i^{1/3^j} = g(b,e)^{1/3^j} \pmod{n'},$$

claiming that this is a signed minor term for an unused 2^j denomination. The bank has no trace of the appropriate b value and would grant the refund. Fortunately, this would not be in Alice's interest, since she would get a smaller refund than she is entitled to.

The last $k - j$ major terms prevent Alice from using the check more than once, Even if the purchase amount is exactly the same. As in section one, the shopkeeper could present a random challenge or every shopkeeper has a probe sequence for these $k - j$ terms chosen from a code with large Hamming distance.

Alice does however, have a good chance of successfully cheating the bank with respect to the refund. All she needs is two unrelated major and minor terms. Still, this type of cheating is far less dangerous than having an open check that can be used over and over again. The bank could penalize Alice whenever it detects an attempt at

cheating, negating Alice's expected profit from cheating attempts. A variation would allocate two major terms per denomination, making the probability of cheating much smaller.

4. Blacklisting Withdrawals

It may be desirable that if Alice uses a coin twice then the bank can blacklist all of the coins Alice has withdrawn. Obviously, this means that all her coins must be related in some manner. The idea is to encrypt some redundancy in Alice's "random" choices; this redundancy can be recognized only when Alice spends a coin more than once. Alice's privacy is thus protected only computationally, *not* unconditionally.

Consider the basic scheme: Alice sends the bank k blinded candidates of the form $r^3 f(g(a,c), g(a \oplus (u \| (v+i)), d))$ where a, c and d are chosen at random by Alice, v is Alice's counter, i is the candidate serial number and u is Alice's account number. We modify the protocol so that Alice generates b electronic coins simultaneously.

Alice sends the bank bk blinded candidates as a matrix

$$\begin{pmatrix} B_{11} & B_{12} & \dots & B_{1k} \\ B_{21} & B_{22} & \dots & B_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ B_{b1} & B_{b2} & \dots & B_{bk} \end{pmatrix}.$$

The bank ask's to see $k/2$ *columns* in their entirety. Each B_{ij} should be of the form

$$r_{ij}^3 f(g(a_{ij}, c_{ij}), g(a_{ij} \oplus (u \| k_j \| (v + ki + j)), d_{ij})).$$

Alice chooses r_{ij}, a_{ij} at random per blinded term and chooses k_j at random per column.

Let $\{h_l\}$ be a family of one-way functions. Each c_{ij} is of the form $h_{l_j}(c'_{ij}) \| c'_{ij}$; each d_{ij} is of the form $h_{l_j}(d'_{ij}) \| d'_{ij}$. Alice chooses c'_{ij} and d'_{ij} at random per blinded term.

The bank can easily verify that each of the $k/2$ columns it sees is of the proper form. For notational simplicity, we assume that the bank asks to see columns $k/2+1, \dots, k$. The bank then supplies Alice with b products

$$P_i = \prod_{1 \le j \le k/2} B_{ij}^{1/3} \pmod{n}$$

and charges her account b dollars.

Alice can then easily extract

$$C_i = \prod_{1 \le j \le k/2} f(g(a_{ij}, c_{ij}), g(a_{ij} \oplus (u \| l_j \| (v + ki + j)), d_{i,j}))^{1/3} \pmod{n}, for 1 \le i \le b.$$

Alice also arranges the factors into lexicographic sequence.

These coins are used exactly as in the basic scheme, except that the shopkeeper has the set of blacklisted indices L. If the merchant sends $e_j = 1$, then Alice must reveal the appropriate a, c and y. The shopkeeper computes $f(g(a,c),y)$ and checks that $c = c''\|c'$ does not satisfy $c'' = h_l(c')$ for all $l \in L$. Similarly, if the shopkeeper sends $e_j = 0$, then Alice must reveal the appropriate $x, a \oplus (u\|h_j\|(v + ki + j))$ and $d = d''\|d'$. Again, the shopkeeper checks that $d'' \neq h_l(d')$ for all $l \in L$.

If Alice uses *any* coin more than once then the bank adds the appropriate revealed k_j's to the blacklist supplied to the merchants.

5. Further Work

In forthcoming work, Chaum and Impagliazzo investigate formal requirements for the function f and den Boer has proposed suitable g's whose security is reducible to factoring or to discrete log. A good deal of progress has been made towards establishing the overall security of similar protocols [CE87]. Formal proofs for the protocols of this paper, however, remain an open challenge.

Acknowledgements

It is a pleasure to thank Russell Impagliazzo for inspiring this work and Eugene van Heigst for making several helpful comments.

References

[BC86a] Brassard, G. and C. Crépeau, Zero-knowledge simulation of Boolean circuits, presented at Crypto '86, 1986.

[BC86b] Brassard, G. and C. Crépeau, Zero-knowledge simulation of Boolean circuits, Proc. of 27*th* Symp. on Foundations of Computer Science, 1986.

[BCC] Brassard, G., Chaum D., and C. Crépeau, Minimum Disclosure Proofs of Knowledge. Center for Mathematics and Computer Science, Report PM-R8710, December 1987.

[C85] Chaum, D., Security without identification: transaction systems to make big brother obsolete, Comm. ACM 28, 10 (October 1985).

[C86] Chaum, D., Demonstrating that a Public Predicate can be Satisfied Without Revealing Any Information About How, presented at Crypto '86, 1986.

[C88] Chaum, D., Privacy Protected Payments: Unconditional Payer And/Or Payee Untracability, Smartcard 2000, North Holland, 1988.

[CE] Chaum, D. and J.H. Evertse, Showing credentials without identification: signatures transferred between unconditionally unlinkable pseudonyms, Proceedings of Crypto '86, Springer-Verlag, 1987.

[CDG] Chaum, D., I. Damgaard, and J. v.d. Graaf, Multiparty computations ensuring the privacy of each party's input and the correctness of the result, Proceedings of Crypto 87, Springer-Verlag, 1988.

[GMR] Goldwasser, S., S. Micali, and C. Rackoff, The Knowledge Complexity of Interactive Proof Systems, Proc. of 17*th* ACM symp. on Theory of Computation, 1985.

[GMW86a] Goldreich, O., S. Micali, and A. Wigderson, How to prove all NP-statements in zero-knowledge, and a methodology of cryptographic protocol design, presented at Crypto '86, 1986.

[GMW86b] Goldreich, O., S. Micali, and A. Wigderson, Proofs that yield nothing but their validity and a methodology of cryptographic protocol design, Proc. of 27*th* Symp. on Foundations of Computer Science, 1986.

[R77] Rabin, M.O., Digitalized signatures, in Foundations of Secure Computation, Academic Press, NY, 1978.

[RSA] Rivest, R.L., A. Shamir, and L. Adleman, A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, CACM 21, 2 (February 1978).