HEWLETT PACKARD

Blind Certification of Public Keys and Off-line Electronic Cash

Wenbo Mao
Networked Systems Department
HP Laboratories Bristol
HPL-96-71
May, 1996

blind certification of public keys, network directory services, off-line electronic cash, one-time signature, prevention of double spending

We present a method for blind certifying end-users' public keys and its application in off-line electronic cash. A blind certificate of a public key is similar to an ordinary public-key certificate; however, the identity of the certificate holder is concealed under the key certified. A digital signature supported by a blind certificate can be verified without identifying the signer. The technique finds a good application in electronic cash. Using a one-time signature scheme during the payment time a spender is required to generate a payment signature supported by a valid blind certificate. Spending a coin once the spender remains anonymous whereas double spending will lead to discovery of the spender's private key and thereby her/his identity. The standard network directory services for certificate revocation (e.g. ISO/IEC X.509 framework) allows prompt revocation of the responsible blind certificate, hence prohibiting any further spending.

Internal Accession Date Only

Blind Certification of Public Keys and Off-Line Electronic Cash

Wenbo Mao
Hewlett-Packard Laboratories
Filton Road
Bristol BS12 6QZ
United Kingdom
wm@hplb.hpl.hp.com

May 1, 1996

Abstract

We present a method for blind certifying end-users' public keys and its application in off-line electronic cash. A blind certificate of a public key is similar to an ordinary public-key certificate, however the identity of the certificate holder is concealed under the key certified. A digital signature supported by a blind certificate can be verified without identifying the signer. The technique finds a good application in electronic cash. Using a one-time signature scheme during the payment time a spender is required to generate a payment signature supported by a valid blind certificate. Spending a coin once the spender remains anonymous whereas double spending will lead to discovery of the spender's private key and thereby her/his identity. The standard network directory services for certificate revocation (e.g., the ISO/IEC X.509 framework) allows prompt revocation of the responsible blind certificate and hence prohibiting any further spending.
Keywords: Blind certification of public keys, Network directory services for public-key certification and certificate revocation, Off-line electronic cash, Prevention of double spending, One-time signature, Schnorr's scheme, Internet electronic commerce.

1 Introduction

Chaum's invention of blind signature techniques [8, 10] sets an important milestone for electronic commerce based on cash transactions. After Chaum's original idea, the subject of electronic cash has been widely studied and many schemes proposed to tackle various unsolved problems, e.g., [3, 4, 5, 6, 7, 9, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 23, 24, 25]. Early schemes for off-line cash (e.g., Chaum, Fiat and Naor [15], Hayes [21], Okamoto and Ohta [25]) are notoriously inefficient as a result of using "cut-and-choose" techniques in cash withdrawal as well as in payment phases in order to thwart cheating (though the method of Okamoto and Ohta [25] includes a binary-tree technique for an efficient representation of coin division into smaller denominations.) Franklin and Yung first introduced a "line method" [19, 20] based on the Diffie-Hellman problem and set off a promising approach to avoidance of cut-and-choose. More

efficient "single-term" coins using no cut-and-choose were subsequently achieved by Brands [4] and by Ferguson [18]. Brands further generalised the line method into a "representation of Diffie-Hellman problem in groups of prime orders" [6] which can be used to design an electronic wallet with an observer nested in. Eng and Okamoto combined the Brands' representation problem with the binary-tree method for an improved efficiency of divisible coins [17]. Recently, Okamoto proposed an idea of "electronic license" [24]. Using a bit-commitment method, a bank can give its customer (an account holder) an "electronic license" which tells a payee the method to verify money issued by it. An electronic license contains a blinded composite number whose factorisation reveals the identity of a bank account holder. The factorisation can be computed if double spent coins are deposited (with the license).

It is our understanding that all of the previous off-line cash techniques known to us have a common problem which can be referred to as an inability to swiftly stop an identified double spender from further spending. In these schemes, during the payment time the payee only check the format legitimacy of coins rather than checking whether any of them has been spent before. Proposals using tamper-resistant devices or observers (e.g., [4, 5, 6, 16]) only intend to make double spending difficult whereas once a device has been successfully opened there is no any mechanism to stop subsequent multiple spendings. Publishing a blacklist of double spenders does not solve the problem because the spender enjoys anonymity during the payment time. The problem is serious indeed considering cash's universal usability and an expected determination of a double spender to commit a mass spending once (s)he starts it.

In this paper we propose a *blind certification technique* that provides a solution to this problem.

A *blind certificate* of a public key is similar to an ordinary public-key certificate, however the identity of the certificate holder is concealed under the key certified. Thus, a signature supported by a valid blind certificate can be verified without identifying the signer. In addition, once a blind certificate has been issued, the link between the certificate and its holder is lost. Therefore the certification authority (CA) cannot be used for the purpose of linking a blind certificate and a person. To be used in conjunction with a one-time signature scheme, the blind certification technique finds a good application in off-line electronic cash. During the payment time the spender is required to produce a one-time signature supported by her/his blind certificate. Spending a coin once the spender remains anonymous whereas double spending will lead to discovery of the spender's private key and thereby her/his identity.

Thanks to the ready availability of the standard network directory services for certificate and public key revocation (following the standard setting for public-key certification and certificate revocation, e.g., in the ISO/IEC X.509 framework, version 3 [22]), it is now easy to swiftly stop any further spending by an identified double spender. Once a double spending is detected, the responsible public key and the blind certificate will be revoked promptly upon showing the matching private key to the CA in charge. The revocation can be in the form of publishing the responsible public key onto the certificate revocation list (CRL) which can be made available world wide. A variation of the CRL (called *delta-CRL*, assumed to be a small file for a short period of time, see Section 12.6 of [22]) can be periodically downloaded by world-wide merchants to update their local copies of CRLs. Thus, shortly after a double spending, shops all over the

world will reject any coin from the double spender including unspent valid coins. Rejecting valid coins from a double spender strongerly deters double spending in the first place. The deterrence is strong in another aspect: a new blind certificate for an identified double spender can be sufficiently expensive so that the price far outweighs the benefit to double spend a limited amount of money before a swift revocation of the responsible public key.

Being able to swiftly stop double spending means that the cash scheme to be proposed need not depend on a tamper-resistant device for the purpose of double-spending prevention. Of course, using a hardware device will be helpful in terms of protecting end-user's private key and of preventing accidental human errors. However note that unlike the "observer" techniques in some previous cash schemes (e.g., [4, 5, 6, 16]), the end-user in our technique does not hold any system system-wide secret and therefore if hardware devices are used, they need not be really tamper-resistant as it is pointless for the holder to tamper the device.

The cash scheme based on the blind certification technique will have another new feature that is not available in all of the previous schemes for off-line cash: there is no need to encode any withdrawers' bank-account information into coins. The advantage of this feature includes: cash usable by non-bank-account holders, cash available ubiquitously, and very small datasize for coin representation (due to total avoidance of cut-and-choose for withdrawal and payment).

Unlinkability is a service stronger than anonymity. (Ferguson differentiates the two notions by *privacy* and *fake privacy* [18]). Rigorously speaking, unlinkability means an inability to determine whether two coins have been spent by the same person (without knowing the person's identity). A weaker meaning is that a person's spending pattern cannot easily be determined (usually by the bank). A unlinkability service (even in the weaker sense) is desirable because in reality a spender's identity may be accidentally disclosed outside the electronic cash mechanism, and without such a service, a person's spending pattern in the history or in the future may be known once her/his identity is accidentally disclosed (a vulnerability in e.g., the scheme of [24]). We will reason that the blind certification technique allows us to achieve a practical quality of unlinkability: it requires an impractically large scale of collusion to determine a person's spending pattern.

The remainder of this paper is organised as follows. The blind certification technique is devised in In Section 2. In Section 3 we introduce a one-time signature technique based on Schnorr's scheme. In Section 4 the signature scheme will be used in conjunction with the blind certification technique to demonstrate the working principle of an electronic cash technique. The security of the cash technique will be analysed in Section 5. Finally, Section 6 concludes the work of this paper.

2 Blind Certification of Public keys

Similar to an ordinary certificate for an end-user's public key, a blind certificate is a digital signature created by an CA to combine a public key and its holder's identity. Nevertheless, the identity of a blind certificate's holder is concealed. Using Chaum's blind signature technique [8] in conjunction with a cut-and-choose method we can devise a protocol for the CA to issue

a blind certificate to Alice for certifying her public key. Here we present a technique based on the RSA crypto-algorithm.

Let (K_{CA}, M_{CA}) be the RSA public key of the CA for certification purposes. To run *certificate issuing protocol*, Alice will prepare N candidate private/public key pairs (s_i, v_i) for $1 \leq i \leq N$. One pair will be chosen out of these N candidates upon termination of a run of the protocol and the chosen key pair will be denoted as (s, v). Further, let $E_s()$ be a symmetric encryption algorithm keyed by the secret key s (here s coincides to, or is truncated from Alice's private key, more details see below); $H()$ be a secure one-way hash function (e.g., SHA [2]); and $\|$ be string concatenation. For $1 \leq i \leq N$, Alice will prepare and send the following N pieces of data to the CA:

Step 1. $A \rightarrow CA:\ H(s_i),\ E_{s_i}(b_i),\ b_i^{K_{CA}} * (H(v_i, E_{s_i}(A)) \| E_{s_i}(A)) (\mathrm{mod} M_{CA})$

Here A denotes Alice's identity; each b_i is called *blinding coefficient* which is a random number chosen from $Z^*_{M_{CA}}$. Upon receipt these data, the CA will randomly choose $1 \leq j \leq N$ and:

Step 2. $CA \rightarrow A:\ j$

Alice then discloses $N-1$ pairs (s_i, v_i) for $1 \leq i \leq N$ and $i \neq j$:

Step 3. $A \rightarrow CA:\ s_i, v_i$

The CA can verify the correctness of the respective $N-1$ pieces of data received in Step 1, which are not indexed by j, to see if Alice has honestly encoded her identity in these data. If the checking passes, then the CA will assume that the remaining item (indexed by j) is also correct and will sign it. Alice has 1 in N chance to succeed a cheating, i.e., to let the CA issue her a blind certificate which encodes a wrong identity. The probability to succeed the cheating is sufficiently small if N is sufficiently large. Below we will further see that even Alice does succeed a cheating, she will still be identified once she uses the blind certificate in a dishonest way.

The CA's signature on the third chunk of the remaining item will yield the following quantity (denoting b_j by b and (s_j, v_j) by (s, v)):

$$(b_j^{K_{CA}} * (H(v_j, E_{s_j}(A)) \| E_{s_j}(A)))^{K_{CA}^{-1}} (\mathrm{mod} M_{CA})$$
$$= b * (H(v, E_s(A)) \| E_s(A))^{K_{CA}^{-1}} (\mathrm{mod} M_{CA})$$

Step 4. $CA \rightarrow A:\ b * (H(v, E_s(A)) \| E_s(A))^{K_{CA}^{-1}} (\mathrm{mod} M_{CA})$

By dividing b into the data received, Alice obtains the following blind certificate:

$$Blind\_Cert_A = (H(v, E_s(A)) \parallel E_s(A))^{K_{CA}^{-1}} (\mathrm{mod} M_{CA})$$

The certificate and the public key v can be validated using the CA's well-known public key. The CA will archive the following data:

$$Archive_A = H(s),\ A,\ E_s(b),\ b^{K_{CA}} * (H(v, E_s(A)) \parallel E_s(A)) (\mathrm{mod} M_{CA})$$

The property of the blind signature ensures that it is computationally infeasible to find any verifierable relationship between $Archive_A$ and $Blind\_Cert_A$. We assume that the CA is a competent party to use appropriate auditing measures to prevent the data $Archive_A$ from being altered or deleted. Such a protection is necessary, however, is out of the scope of this paper.

The blind certification technique will be used in conjunction with a special signature scheme (to be introduced in the next section) to serve the following function: honestly using the system the user will be served with anonymity protected; abusing the system will lead to discovery of the user's private key s and thereby her/his identity by decrypting the archived data. Upon identification, the matching public key and the certificate will be revoked promptly by publishing them in the CRL. Earlier we have mentioned that even Alice does succeed a cheating by having encoded a wrong identity into a blind certificate, the success will not help her to gain a very clear advantage. Upon her abusing the system that leads to discovery of her private key, the matching public key can still be revoked regardless of whether or not the revealed identity matches the cleartext A in $Archive_A$.

For simplicity, we have omitted presentation of some implementation related information. For instance, a blind certificate should contain information pointing to the network directory services (e.g., the trust chains and the CRL locations); also, a private key s should contain a subfield (e.g., two least significant bytes) to uniquely point to the CA who archives the data "$H(s), \cdots$" so that, with a revealed private key s, the dishonest user can be identified from the right CA. Also we assume that the cryptographic codings contain properly redundant information needed to foil various forms of brute-force searching targeted at the private key s.

3 A One-time Signature Scheme

The one-time signature technique is based on Schnorr's signature scheme. We start by presenting Schnorr's original scheme [27, 28, 29].

In Schnorr's scheme, users in the whole system can share some public values as part of their public keys. First, choose two large primes, p and q where p is sufficiently large (e.g., $p > 2^{512}$) such that the discrete logarithm problem in Z_p is intractable; q is also large (e.g., $q > 2^{140}$) and $q|(p-1)$. Then, choose a number $a \in Z_p \setminus \{1\}$ which generates a group with order q; namely, q is the smallest positive integer satisfying $a^q \equiv 1 (\mathrm{mod} p)$. The group generator a can be computed as the $(p-1)/q$th power of a primitive element modulo q. It will be assumed that

all parties in the system share these numbers. To generate a particular private/public key pair, Alice chooses a random number s for $1 \leq s \leq q$. This is her private key. Then she calculates

$$v := (a^{-s} \bmod p) \tag{1}$$

The result v is Alice's public key. Schnorr's scheme uses a secure one-way hash function. Let $h()$ denote such a function which maps from the integer space to Z_q. To sign a message m, Alice picks a random number $r \in Z_q$ and does the following computations:

$$x := (a^r \bmod p) \tag{2}$$

$$e := h(m, x) \tag{3}$$

$$y := ((r + se) \bmod q) \tag{4}$$

The signature on the message m is the pair (e, y). We will call the other two quantities r and x, *secret* and *public signature generators* (or SSG, PSG for short), respectively. To verify the signature, Bob computes:

$$z := (a^y v^e \bmod p) \tag{5}$$

and tests if $e == h(m, z)$. If the testing is true, Bob accepts the signature as valid.

Schnorr's signature scheme gets its security from the difficulty of calculating discrete logarithm. The difficulty means that the private key s cannot be easily derived from the public key v from their relation in (1). Similarly, the SSG r cannot be easily derived from the non-secret number z from their relation in (2) (z is equal to the PSG x). If SSG r can easily be discovered, then the private key s can easily be derived from (4).

Besides relying on the difficulty of discrete logarithm, the security of the one-way hash function $h()$ also plays an important role. The security is in terms of the infeasibility to invert the function, and to find two input values $x \neq x'$ such that $h(x) = h(x')$. When Bob verifies the signature, he knows from the property of the hash function that, without knowing Alice's private key, it is computationally infeasible to create the consistency among the numbers e, z and m which are related under the hash function used.

Note that the SSG r must be treated as one-time material. It must not be used more than once to generate different signatures. Assume that Alice has used an SSG r before to sign a message m and now she re-uses it to sign a different message m'. Let (e, y) and (e', y') be the respective signatures. Now that $m' \neq m$, from (3) and the property of the hash function we know with an overwhelming probability that $e' \neq e (\bmod q)$. With the two signatures, Bob can compute Alice's private key s by subtracting two instances of (4) and obtain

$$s = (\frac{y - y'}{e - e'} \bmod q) \tag{6}$$

When creating a new signature, as long as Alice always choose a new SSG at random from Z_q, then subtraction of (4) will only result in

$$y - y' \equiv r - r' + s(e - e') \ (\bmod q)$$

Here the value $r - r' \not\equiv 0 (\mathrm{mod} q)$ remains to be a secret that protects the private key s just in the same way as in Schnorr's scheme. More precisely, Alice should not use an SSG which is related to old SSG's in any known algorithmic way. As long as this precaution measure is taken, no computationally feasible method is known to derive the private key from different instances signatures. In fact, the digital signature standard (DSS) proposed by NIST [1] uses essentially the same principle to protect the signer's private key.

We can employ the property illustrated in (6) to identify a user who has used certain data more than once. The idea is to let the user sign the data as a condition to use the data and the signature must be generated in such a manner that the verifier can tell whether the signer has correctly complied a specified procedure. Note that to produce a signature supported by a blind certificate need not identify the signer. Electronic cash forms a good example of such data. A coin (blindly signed by the bank) can be constructed to contain a PSG x. During the payment time Alice must sign the coin for the merchant to verify using her blind certificate. The coin will not be accepted if Alice's signature has used a wrong PSG (the merchant sees z in (5) to be different from x in the coin) even if the signature is valid in Schnorr's scheme. Thus, Alice cannot double spend a coin without disclosing her private key.

We point out that it is impossible for Alice to find two SSG's $r \not\equiv r' (\mathrm{mod} q)$ that map to the same PSG x (i.e., $x = a^r = a^{r'} (\mathrm{mod} p)$). Being able to do so will allow Alice to cheat the bank with $r \not\equiv r' (\mathrm{mod} q)$, and also to cheat the merchant as if a "correct" PSG has been used. The impossibility can be demonstrated as follows. From $a^r = a^{r'} (\mathrm{mod} p)$, we have $a^{r-r'} = 1 (\mathrm{mod} p)$. But remember that a is a group generator with order q which means q is the smallest positive integer satisfying $a^q = 1 (\mathrm{mod} p)$. So it has to be $r \equiv r' (\mathrm{mod} q)$.

4 A Simple Off-Line Electronic Cash Scheme

We devise a simple off-line electronic cash scheme. The scheme consists of three protocols: withdrawal, payment and deposit.

4.1 Withdrawal

Alice can withdraw a coin by running a withdrawal protocol with a bank (any bank). The coin will be blindly signed by the bank to worth a specified value and this can be validated by any receiver. Let B denote the bank; (K_B, M_B) be the bank's RSA public key; b be a blinding coefficient randomly chosen by Alice from $Z^*_{M_B}$, $f()$ be a secure one-way hash function, x be a PSG pre-computed by Alice and v be Alice's public key in Schnorr's scheme. The withdrawal protocol can be as follows.

Step 1. $A \rightarrow B$: $request$, $b^{K_B} * f(x, v) (\mathrm{mod} M_B)$

Step 2. $B \rightarrow A$: $(b^{K_B} * f(x, v))^{K_B^{-1}} (\mathrm{mod} M_B)$

Alice obtains the coin by dividing the blinding coefficient b into the data received:

$$Coin = f(x, v)^{K_B^{-1}} (\mathrm{mod} M_B)$$

The message "*request*" in Step 1 represents a (credit or debit) transaction request. It instructs the bank a method to obtain money from Alice. It is not necessary for Alice to withdraw money from her own bank; for instance, the *request* in the protocol can be a result of another fund transfer protocol. The withdrawal protocol does not resort to any cut-and-choose technique. For her own interest, Alice should not construct an invalid coin, such as to have encoded an invalid PSG x (e.g., a replayed PSG, or not knowing the matching SSG), or an invalid public key v (e.g., uncertified, or not knowing the matching private key) into the coin; or else the money is wasted.

4.2 Payment

When paying *Coin* to a merchant, Alice must sign a spending signature using the PSG x integrated in *Coin*. The spending signature should include the merchant's identity and a timestamp stating the spending time. Let M be the merchant's identity, and *DateTime* be a timestamp. The to-be-signed message should be "*Coin*, M, *DateTime*". Following (3) and (4), the spending signature is a pair (e, y) where

$$e := h(Coin, M, DateTime,\ x) \quad \text{and} \quad y := ((r + se) \bmod q)$$

It suffices to use the following single step to specify the payment protocol:

$$A \rightarrow M : Coin,\ M,\ DateTime,\ e,\ y,\ v,\ Blind\_Cert_A$$

Before verifying the spending signature, the merchant should first validate the public key and the supporting blind certificate. The validation includes to check if the public key has been revoked. A revoked key should appear in his local CRL as a result of his periodical downloading the delta-CRL from the network directory services to update his local copy of CRL (see Section 12.6 of [22]). If v is properly backed by $Blind\_Cert_A$ and is not in the CRL, the merchant will carry on to verify the spending signature. This is to compute z from a, v, e, y as in (5), and test if

$$e == h(Coin, M, DateTime,\ z)$$

He should also check the correct use of the PSG and the public key. The correctness of these values means that z and v can be hashed to $f(x, v)$ in *Coin*. Permitting to use a wrong PSG or an invalid public key will put the merchant in trouble if Alice later double spends the coin.

4.3 Deposit

In an off-line time later, the merchant will redeem *Coin* by depositing it to the bank:

$$M \rightarrow B : Sign_M(Coin, M, DateTime, E_M(z), e, y)$$

We call these data *coin-deposit.* Here, $Sign_M()$ denotes a digital signature of the merchant and $E_M(z)$ means that the merchant encrypts z using his own public key. (Usual implementations of digital signatures use one-way hash function and so $Sign_M(\cdots, E_M(m), \cdots)$ will not reveal the encrypted message m.) The merchant's signature on the coin-deposit means that the merchant has properly dealt with the data in the payment and the deposit protocols. The bank cannot alter the coin-deposit (e.g., to frame the merchant).

5 Analysis

Now we examine the security of the electronic cash scheme.

5.1 Strong anonymity

First, we assume that Alice does not double spend her coins. Her anonymity of using the coins will be protected. This is because no data in the payment and in the deposit protocols contains any information about her identity. The anonymity service is in a strong sense in that, collusion among banks, all merchants and CAs will not result in any computationally feasible way to identify the spender of a coin.

5.2 Practical unlinkability

A pragmatic unlinkability service is supplied. The service means that it is impractical for any party in the system to determine an anonymous spender's spending pattern. By impracticalness, we will reason that in order to determine the spending pattern of an anonymous spender, the bank and merchants in the system have to collude in large scales. Note that because money will eventually converge to the bank, the bank is in a better position than any merchant to link a large volume of coins. Our analyses in the unlinkability will therefore be focused on the bank, with and without the help from merchants.

First of all, it is obvious that if public keys and/or the supporting blind certificates are deposited together with coins, then coins are partitioned by public keys and/or blind certificates and all coins in the same partition are spent by the same person. Depositing public keys or blind certificates is regarded a collusion. Slightly less obvious is that the PSG value z, if deposited, can also be used to derive the public key v. This is because of the following congruence:

$$v^e \equiv z/a^y (\mathrm{mod} p) \tag{7}$$

Once v^e is known, it is easy to reveal v as

$$v := (v^{ed} \bmod p) \quad \text{where} \quad ed \equiv 1 (\mathrm{mod} p - 1) \tag{8}$$

Without giving these values to the bank, it is computationally infeasible for the bank to link coins that have deposited. (It suffices for a merchant to be non-collusive if he simply forgets the anonymous spender's public key, the blind certificate and the PSG z once after the coin has been accepted.) Each coin is a function of a one-time random PSG, so is each spending signature. Thus, no pair of undouble-spent coin-deposits will give any information whatsoever about their relationship. Brute-force searching through the public-key space, e.g., using a candidate public key v and (5) to get a candidate PSG z followed by checking if they can be hashed to $f(x, v)$ in *Coin*, is intractable as the searching has to go through the vast space Z_p, unless the bank has acquired a sufficiently large number of public keys of the users in the system (which form a trivially small subset of the whole public-key space). However to collect public keys requires a large scale collusion among the banks and the merchants. Brute-force searching through the PSG space for mathing $E_M(z)$ (the resultant value will allow to compute a respective public key using (7) and (8)) is samely infeasible as that of searching the public-key space, and the searching can also be thwarted by adding appropriate redundancy into the encryption $E_M(z)$.

Finally we point out that even the bank has successfully collected a large volume of data needed to investigate an anonymous person's spending pattern, the data are only good for knowing the person's spending *history*. Linking future coins requires further collusion. Indeed, the scale of collusion needed is related to the number of coins deposited. The necessity for maintaining a long-term and large-scale collusion forms the foundation for us to claim the impracticability of the collusion, or in other words, the pragmatic quality of the unlinkability service of our technique.

5.3 Correctness

Now we look at the difficulty for various parties to defraud. Assume that the bank sees duplicated copies of *Coin*. This may be resulted from either (i) Alice's double spending, or (ii) the merchant's replay or depositing of bad data, or (iii) a collusion between Alice and the merchant.

Case (i) Alice double spends. The differences in either M or *DateTime* will result in two pairs of spending signatures (e, y) and (e', y') where $e \not\equiv e' (\mathrm{mod} q)$ (and hence $y \neq y'$) with an overwhelming probability. These two pairs will suffice the bank to discover Alice's private key s using (6), and further obtain her public key v from (1).

The bank can see the correctness of the revealed keys by re-verifying the two spending signatures as the merchants have done. Any incorrectness in the re-verification indicates either a collusion between Alice and the merchant(s), or a fraudulent merchant. These will be dealt with in Cases (ii) and (iii). Assume that the reverification of the spending signatures passes. Now the bank can identify Alice by showing the revealed private/public key pair (s, v) to the appropriate CA. (The private key s contains a subfield that uniquely points to the CA who has issued the blind certificate to Alice.) Upon seeing the revealed key pair, the CA will search the archive for matching value $H(s)$ and the associated item $Archive_A$. Once a match is found, the public key v will be revoked promptly and unconditionally (by publishing it onto the delta-CRL). Alice's identity is also revealed as a result.

Case (ii) The merchant replays data or deposits bad data. Because the merchant is unable to generate a valid spending signature using other peoples coins, double depositing coins is confined to the following uninteresting way: the merchant simply replays all messages in the deposit protocol. It is easy for the bank to discover the replay and thereby only one instance of deposit will be redeemed.

Note that since the merchant is required to digitally sign each coin-deposit, depositing incorrect data containing gibberish as "spending signatures" will lead to identifying the merchant as fraudulent as a result of bank finding incorrect spending signatures during their re-verification. In Case (iii) we will see a method to distinguish an honest merchant from bad ones.

Case (iii) Alice and the merchant collude. A collusion will make sense only if it does not lead to identifying Alice. Feasible ways to achieve this include that the merchant permits using incorrect public keys (uncertified or not matching v in *Coin*), or incorrect PSG's (not matching x in *Coin*). For instance, in the case of permitting using incorrect keys, a coin can be double spent by different people, or by the same person who holds different (certified or not) public keys.

Firstly, we assume that the merchant permits using a uncertified public key when verifying the spending signature; namely, the public key used is not supported by a valid blind certificate. This collusion will be discovered because the correctly revealed private key (assuming in a valid format leading to a known CA) will not find an archived item $Archive_X$ for any X. Such a public key will also be published in the CRL to stop any further collusion.

In other circumstances, upon seeing duplication of *Coin*, the bank's computation using (6) will not reveal a correct private key. For instance, assume that two spending signatures (e_1, y_1) and (e_2, y_2) have been generated by two different key pairs where the private keys are s_1 and s_2, respectively. Then, using (6) will result in the following value:

$$s' = (\frac{s_1 e_1 - s_2 e_2}{e_1 - e_2} \bmod q)$$

Similarly, assume a merchant permits using an incorrect PSG which is mapped from a wrong SSG $r' \not\equiv r (\bmod q)$ where r may or may not be a valid SSG. Then (6) will disclose the following value:

$$s' = (\frac{s(e_1 - e_2) \pm (r' - r)}{e_1 - e_2} \bmod q)$$

Other wrong forms of "private keys" can also be derived by mixed uses of wrong/good keys and wrong/good PSG's. Let v' be the matching "public key" computed from s' using (1). The bank will always re-verify the two spending signatures using the revealed public key v' as the merchant(s) have done during the two runs of the payment protocol. The re-verification will result in inconsistency; e.g., either s' is in an invalid format (does not point to a correct CA), or the two spending signatures $(e_{1,2}, y_{1,2})$ are incorrect regarding the verification key v' used, or the quantities z' and v' cannot be hashed to $f(x, v)$ in *Coin*. (N.B. the re-verification excludes any possibility of mistakenly identifying an innocent user whose private key coincides to s'

because even in such an extremely unlikely case, the "spending signatures" $(e_{1,2}, y_{1,2})$ will be found to be incorrect when verified using v' as they were not created by s'.)

In these situations, the two merchants (let them be M_1 and M_2) will be asked to decrypt $E_{M_1}(z)$ and $E_{M_2}(z)$, respectively, in order to prove their honesty. An honest merchant will be indicated by a z which can derive a public key v using (7) and (8) such that v is not in CRL, and using it the spending signature deposited by the merchant can be re-verified as correct (including that the quantities z, v can be hashed to $f(x, v)$ in $Coin$). The other merchant will be identified as fraudulent.

To this end, we see that the merchant is unable to help Alice to double spend.

It is interesting to point out that, as long as a coin is not to be double spent or double deposited, using invalid public keys or incorrect PSG's or even depositing gibberish spending signatures will not be detected since the bank will not and cannot verify the "spending signature". Indeed, the bank need not be concerned of anything other than double spending.

6 Conclusion

We conclude the work with a summary on the features in the electronic cash scheme that uses the blind certification technique.

An effective solution to stop double spending. This is a unique feature that is not available in all previous off-line electronic cash schemes. With this feature, off-line electronic cash becomes practically usable. The need of using tamper-resistant devices for double-spending prevention is not needed. Of course, using tamper-resistant devices will be undoubtedly helpful in protecting the user's private keys and in preventing accidental human errors.

Strong anonymity for the spender. As long as the spender does not double spend, no party in the system can have any computationally feasible way to identify who has spent the money. The anonymity service is strong in that, collusion among all parties will not achieve a computationally feasible method to compromise the quality of the service.

Practical unlinkability. The bank does not have a computationally feasible way to determine the spending pattern of an anonymous person. This is only true in a weak sense since collusion among the bank and merchants will allow coin linkage. However, collusion of a large scale (proportional to the volume of coin-deposit) is required in order to collect useful information. Furthermore, data gathered in the history are not useful for linking coins to be deposited in the future and further collusion is necessary for that purpose.

System simplicity. The protocols and the underlying mathematics for blind certificate issuing, cash withdrawal, payment and deposit are exceptionally simple. The datasize for representing a coin is trivially small (virtually a blind signature on an image of a hash function). Cash can easily be withdrawn from a foreign bank and can be used by a non-bank-account

holder. We believe that a binary tree technique (e.g., in [25]) or a chained hash technique (e.g., in the form of Payword [26]) can be applied to achieve coin sub-divisibility. The system simplicity demonstrates a readily applicable electronic cash scheme.

References

[1] Proposed Federal Information Processing Standard for Digital Signature Standard (DSS). Federal Register, v.56, n.169, August 1991.

[2] NIST FIPS PUB 180, Secure hash standard. National Institute of Standards and Technology, U.S. Department of Commerce, DRAFT, April 1993.

[3] J.-P. Boly et al. The ESPRIT Project CAFE — High Security Digital Payment Systems. In *Computer Security — ESORICS'94 (LNCS 875)*, pages 217–230. Springer-Verlag, 1994.

[4] S. Brands. Untraceable off-line cash in wallet with observers. In *Advances in Cryptology — Proceedings of CRYPTO'93 (LNCS 773)*, pages 302–318. Springer-Verlag, 1993.

[5] S. Brands. Electronic cash on the internet. In *Proceedings of the Internet Society 1995 Symposium on Network and Distributed System Security*, 1995.

[6] S. Brands. Off-line electronic cash based on secret-key certificates. Technical Report: CS-R9506, 1995.

[7] J. Camenisch, J-M. Piveteau, and Stadler M. An efficient electronic payment system protecting privacy. In *Computer Security — ESORICS'94, (LNCS 875)*, pages 207–215. Springer-Verlag, 1994.

[8] D. Chaum. Blind signatures for untraceable payments. In *Advances in Cryptology — Proceedings of Crypto'82*, pages 199–203. Plenum Press, 1983.

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

[10] D. Chaum. Blind signatures systems. U.S. Patent No 4,759,063, July 1988.

[11] D. Chaum. Privacy protected payments: Unconditional payer and/or payee untraceability. In *Smartcard 2000*. North Holland, 1989.

[12] D. Chaum. Online cash checks. In *Advances in Cryptology — Proceedings of EUROCRYPT'89 (LNCS 434)*, pages 288–293. Springer-Verlag, 1990.

[13] D. Chaum. Achieving electronic privacy. *Scientific American*, pages 96–101, August 1992.

[14] D. Chaum, B. den Boer, E. van Heyst, S. Mjolsnes, and A. Steenbeek. Efficient offline electronic checks. In *Advances in Cryptology — Proceedings of EUROCRYPT'89 (LNCS 434)*, pages 294–301. Springer-Verlag, 1990.

[15] D. Chaum, A. Fiat, and M. Naor. Untraceable electronic cash. In *Advances in Cryptology — Proceedings of CRYPTO'88 (LNCS 403)*, pages 319–327. Springer-Verlag, 1990.

[16] D. Chaum and T. Pedersen. Wallet databases with observers. In *Advances in Cryptology — Proceedings of CRYPTO'92 (LNCS 740)*, pages 89–105. Springer-Verlag, 1992.

[17] T. Eng and T. Okamoto. Single-term divisible electronic coins. In *Advances in Cryptology — Proceedings of EUEOCRYPT'94 (LNCS 950)*, pages 306–319. Springer-Verlag, 1995.

[18] N. Ferguson. Single term off-line coins. In *Advances in Cryptology — Proceedings of EUROCRYPT'93 (LNCS 765)*, pages 318–328. Springer-Verlag, 1994.

[19] M. Franklin and M. Yung. Towards provably secure efficient electronic cash. Technical Report: TR CUCS-018-92, April 1992.

[20] M. Franklin and M. Yung. Secure and efficient off-line digital money. In *Proceedings of ICALP'93, (LNCS 700)*, pages 265–276. Springer-Verlag, 1993.

[21] B.. Hayes. Anonymous one-time signatures and flexible untraceable electronic cash. In *Advances in Cryptology — Proceedings of AUSCRYPT'90 (LNCS 453)*, pages 294–305. Springer-Verlag, 1990.

[22] ITU/ISO/IEC. Draft Amendment 1 to ITU Rec. X.509 (1993) — ISO/IEC 9594-8: Information Technology — Open Systems Interconnection — The Directory: Authentication Framework, Amendment 1: Certificate Extensions. ISO/IEC JTC 1/SC 21/WG 4 and ITU-T Q 15/7 Collaborative Editing Meeting on the Directory, Ottawa, Canada, July 1995.

[23] G. Medvinsky and B.C. Neuman. NetCash: A design for practical electronic currency on the Internet. In *Proceedings of First ACM Conference on Computer and Communications Security*, pages 102–196. ACM Press, 1993.

[24] T. Okamoto. An efficient divisible electronic cash scheme. In *Advances in Cryptology — Proceedings of CRYPTO'95 (LNCS 950)*, pages 438–451. Springer-Verlag, 1995.

[25] T. Okamoto and K. Ohta. Universal electronic cash. In *Advances in Cryptology — Proceedings of CRYPTO'91 (LNCS 576)*, pages 324–337. Springer-Verlag, 1992.

[26] R.L. Rivest and A. Shamir. Payword and micromint: Two simple micropayment schemes. `http://theory.lcs.mit.edu/~rivest/publications.html`, December 1995.

[27] C.P. Schnorr. Efficient signature generation for smart cards. In *Advances in Cryptology — Proceedings of CRYPTO'89 (LNCS 435)*, pages 239–252. Springer-Verlag, 1990.

[28] C.P. Schnorr. Efficient signature generation for smart cards. *Journal of Cryptology*, 4(3):161–174, 1991.

[29] C.P. Schnorr. A method for identifying subscribers and for generating and verifying electronic signatures in a data exchange system. U.S. Patent No. 4,995,082, February 1991.