Proceedings of Financial Cryptography 2000
(february 21 - 24, 2000, Anguilla, British West Indies)
Y. Frankel Ed. Springer-Verlag, LNCS 1962, pages 259–275.

Self-Scrambling Anonymizers

David Pointcheval

LIENS–CNRS – École Normale Supérieure – France.
E-Mail:David.Pointcheval@ens.fr – http://www.di.ens.fr/~pointche

Abstract. For the two last decades, people have tried to provide practical electronic cash schemes, with more or less success. Indeed, the most secure ones generally suffer from inefficiency, largely due to the use of restrictive blind signatures, on the other hand efficient schemes often suffer from serious security drawbacks. In this paper, we propose both a new tool providing scalable anonymity at a low cost, and a new Internet business: "Anonymity Providers".
Those "Anonymity Providers" certify re-encrypted data after having been convinced of the validity of the content, but without knowing anything about this latter. It is a very useful third party in many applications (*e.g.* for revocable anonymous electronic cash, where a coin would be a certified encryption of the user's identity, such that a Revocation Center, and only it, can recover this identity, if needed).
With this new tool, each user can get the required anonymity level, depending on the available time, computation and/or money amounts. Furthermore, the "Anonymity Provider" may be a new type of business over the Internet, profitable for everybody:
- from the provider point of view as he can charge the service;
- from the user point of view as he can obtain a high level of anonymity at low computational cost. Moreover, a user who does not require anonymity has no extra computation to perform.

This technique is quite efficient because of its "optimistic" orientation: in case of honest use, everything is very efficient. Some slightly more heavy processes have to be performed in case of fraud detection, but with overwhelming tracing success.

Key Words: Electronic Cash, Revocable Anonymity, Designated Verifier Undeniable Signatures, Optimistic Protocols.

1 Introduction

1.1 Background

Recently, electronic commerce and many other applications over the Internet have known a growing activity. However, in order to solve security concerns while providing both flexibility and efficiency, cryptography has a hard task to perform.

Since the Diffie-Hellman paper [17], introducing the concept of public-key cryptography, many tools from the material world have been moved to the electronic one. Among these, most prominently, digital signatures [18, 22] to ensure authentication and non-repudiation of facts or messages and encryption schemes [21] to provide confidentiality (instead of using safe-deposit boxes!). Since the early 80s, Chaum wanted to mimic money [11] and therefore defined the electronic cash notion, originally based on electronic coins and blind signatures [31, 35]. Indeed, this technique helped to define electronic cash schemes that reached a perfect anonymity of transactions, with unlinkability (between two transactions of a same user) and untraceability (between the payer and the payee).

However a crucial problem came from over-spending, which refers to the situation in which a user spends the same coin two or more times. An inherent

quality of digital data is that perfect copies are easy to make; therefore such fraud cannot be avoided, but just detected in the best case. Then, either the detection is done at the spending time which requires the bank to be on-line, or the detection is done later. However, what may be done if the coin is completely anonymous? To address this problem, Chaum, Fiat and Naor [13] used the "cut-and-choose" technique [36] to embed the identity of the user in the coin in such a way that this identity remains perfectly concealed after just one spending but gets revealed after twice.

A new problem later on discovered is the danger of such a perfect anonymity which allows "perfect crime" [41] (without any risk to be caught). Therefore revocable anonymity (after just one spending, or even before any spending) became the new natural approach, giving the control of all privacy issues to a trusted party. Many such schemes were proposed, based on various cryptographic primitives: escrow cash [7, 20] and restrictive/fair blind signatures [6, 8, 9, 37, 38].

1.2 Motivation

Electronic cash is a very crucial topic. However, most of the proposed schemes just rely on heuristic proofs of security and therefore do not formally prevent fraud and counterfeit money. Furthermore, the very few provably secure schemes are either just impractical, or at least very heavy to implement, due to their use of restrictive blind signatures.

However, tools providing anonymity exist: *e.g.* the mix-networks [10, 1, 24, 26, 2, 25] introduced by Chaum, the "crowds" technique [39] suggested by Reiter and Rubin, "magic-ink" signatures [30, 27] proposed by Jakobsson and Yung, which are more like blind signatures. Nevertheless they do not seem to solve all practical issues, from the computational point of view, namely in electronic cash setting. Then new tools would be welcome.

1.3 Outline of the Paper

This paper provides the new notion of "self-scrambling anonymizer" based on "homomorphic electronic coins" together with undeniable signatures [14, 12, 15, 28, 32]. It is therefore rather like mix-networks, using re-encryption techniques together with proofs of equivalence of ciphertexts [25, 29]. Furthermore, it supplies the user with both fully-revocable and scalable anonymity for each coin, depending on the required untraceability and the available computational power/time.

First, some useful building blocks are reviewed. Then, the security model is presented, followed by the intuition behind "self-scrambling", and an informal presentation of the mechanism. A more technical part follows, with a more detailed description of the new tool, together with some security arguments. Finally, we present a candidate based on the famous El Gamal [18] encryption scheme. The security is then proven relative to the Decisional Diffie-Hellman problem [17, 5].

2 Some Building Blocks

Before any technical development, let us review a few well-known cryptographic primitives which will be used in the following.

2.1 Encryption and Semantic Security

To provide anonymity, we will use a public-key encryption scheme with the semantic security notion [21]. The following definitions use some classical notations, but the reader is referred to [3] for more details.

Definition 1 (Encryption Scheme). An *Encryption Scheme* consists of three algorithms: the *key generation* algorithm $\mathcal{K}$ which outputs random pairs of secret and public keys $(\mathsf{sk}, \mathsf{pk})$, the *encryption* algorithm $\mathcal{E}(\mathsf{pk}, m; r)$ which encrypts any message m using a given random tape r and the *decryption* algorithm $\mathcal{D}(\mathsf{sk}, c)$ which inverts the encryption c getting back the plaintext.

Definition 2 (Semantic Security). An encryption scheme $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ is said *Semantically Secure* if given the encryption of one of two chosen messages, the attacker cannot guess the corresponding plaintext. More formally, for any attacker $\mathcal{A} = (A_1, A_2)$,

$$\Pr\left[\begin{array}{l}(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathcal{K}, (m_0, m_1) \leftarrow A_1(\mathsf{pk}) \\ b \stackrel{R}{\leftarrow} \{0,1\}, r \stackrel{R}{\leftarrow} \{0,1\}^\star, c \leftarrow \mathcal{E}(\mathsf{pk}, m_b; r)\end{array} : A_2(c) = b\right] \text{ is negligible.}$$

Example 3. A well-known example is the El Gamal encryption scheme [18]: For a given generator g of a group $\mathcal{G}$, $y = g^x$ is a public key, associated to x.

- The encryption algorithm works as follows:
  $\mathcal{E}(y, m; r) = (g^r, y^r \times m)$ for $m \in \mathcal{G}$ and $r \stackrel{R}{\leftarrow} \mathbb{Z}_{\mathsf{Ord}(g)}$.
- The decryption a given ciphertext (a, b) is just $m = b/a^x$.

2.2 Signature Scheme

In any public key infrastructure, one needs a signature scheme, at least to certify public data, but also objects, messages or facts. For electronic cash, it is needed to certify coins.

Definition 4 (Signature Scheme). A *Signature Scheme* consists of three algorithms:

- the *key generation* algorithm $\mathcal{K}$ which outputs random pairs of secret and public keys $(\mathsf{sk}, \mathsf{pk})$.
- the *signature* algorithm $\mathcal{S}(\mathsf{sk}, m)$ which, on input a message m, returns a valid signature s on it.
- the *verification* algorithm $\mathcal{V}(\mathsf{pk}, m, s)$ which, on input the message m and a signature s, checks whether s is a valid signature or not.

In the following, we will require a secure signature scheme, in the strongest sense: impossibility of an existential forgery even under chosen-message attacks.

Definition 5 (Secure Signature Scheme). A *Signature Scheme* $(\mathcal{K}, \mathcal{S}, \mathcal{V})$ is said *secure* if any attacker $\mathcal{A}$ cannot perform an existential forgery even in an adaptively chosen-message scenario [22], but with a negligible probability: even if the attacker has access to a signer oracle, it cannot produce a valid signature on a new message.

To remain with discrete-log based cryptographic schemes, one can think to the Schnorr-like schemes [40] which derive from interactive zero-knowledge proofs "à la Fiat–Shamir" [19, 34, 35]. Indeed, they have been proven *secure* in the random oracle model [4].

2.3 Designated Verifier Undeniable Signatures

When one uses interactive zero-knowledge proofs, it just convinces the on-line verifier. But the verifier may want to be able to transfer his conviction, with the help of the prover, so that the prover cannot deny his former proof:

Definition 6 (Undeniable Proof Scheme). An *Undeniable Proof Scheme* consists of the following algorithms:

- the *key generation* algorithm $\mathcal{K}$ which outputs random pairs of secret and public keys $(\mathsf{sk}, \mathsf{pk})$.
- the *proof* algorithm $\mathcal{P}(\mathsf{sk}, m)$ which, on input a fact m, returns an "undeniable signature" s on m.
  However this proof "s" does not convince anybody by itself. To get convinced of the validity of the pair (m, s), relatively to the public key pk, one has to interact with the owner of the secret key sk:
- the *confirmation process* $\mathsf{Confirmation}(\mathsf{sk}, \mathsf{pk}, m, s)$ which is an interactive protocol between the signer and the verifier, where the prover (the signer) tries to convince the validity of the pair (m, s).
- the *disavowal process* $\mathsf{Disavowal}(\mathsf{sk}, \mathsf{pk}, m, s)$ which is an interactive protocol between the signer and the verifier, where the prover (the signer) tries to convince that the pair (m, s) is not valid (*i.e.* has not been produced by him).

Both confirmation and disavowal processes are exclusive, which means that the prover cannot succeed in both with non-negligible probability: if he has really produced the signature s, he will be able to confirm but not to deny, and vice-versa.

As any interactive process, confirmation and disavowal can be turned into non-interactive ones, using the Fiat-Shamir's heuristic [19]. But then, after a confirmation, the signature can convince anybody. One way to avoid that is to use a *designated verifier non-interactive proof* where the verifier is the only one to be convinced as he could have produced it.

Many undeniable proofs exist in the literature [14, 12, 15, 32], with various integrations into large applications [23]. Furthermore, some general conversions provides designated verifier signatures [28].

3 Security Model

To provide a clear security model, even if the new notion can be suitable to many other applications, in the following we focus on electronic cash concerns. More precisely, as presented in the introduction, we will formalize the requirements for revocable anonymous electronic cash.

First we introduce the participants. Then we precise the communication model. Thereafter we define the anonymity requirements and the expected specifications for the revocation mechanism. Finally, we precise the diagram of trust.

3.1 Participants

In a classical payment scenario, three people are involved: the bank, the user (a.k.a. consumer) and the shop. The consumer withdraws money from his account in the bank, then he can spend it in the shop who finally deposits it on his own account at the bank.

To satisfy the anonymity properties, some third parties will be involved in our scenario:

- some "Anonymity Providers" (APs in short) will help the user to make transactions anonymous;
- a "Revocation Center" (RC in short) will have the ability to get back the identity of a frauder from a coin or a transaction.

As in any public-key setting, each participant possesses a public-secret key pair certified by a trusted-authority we will not consider anymore. Therefore, we can identify the identity of a participant with his public key. The secret keys of the bank and anonymity providers will be used to certify coins, while the public keys will be involved in the verification process.

3.2 Communication Model

In all the following, no assumption is made about the network, or more formally about the communication channels: any communication is publicly available to anybody. However, we will assume, as usual, that any exchange of data is done in a fair way: when the user correctly asks for a withdrawal to the bank, the bank returns a coin; when the user has paid for a service to a shop, he really receives the service, etc.

3.3 Anonymity

In large scale electronic transaction systems, many informations can be learned about users. More precisely, huge databases about personal profiles could be built. Then anonymity in this domain has been considered as a crucial property [10]. Therefore, two notions of anonymity have been identified:

- unlinkability, which refers to the inability for anyone to link two transactions performed by a same user;

– untraceability, which refers to the inability for anyone to match a transaction with a user.

Furthermore, about such links, two levels of anonymity can be considered:

– strong anonymity: nobody can guess the link, but with negligible probability;
– weak anonymity: some people may know the link, however they are unable to prove it, but with negligible probability.

For example, in our proposal, we will see that strong anonymity is achieved as soon as one participant in the following long list: bank, APs, shop, is honest. Otherwise, only one AP, even a dishonest one, is enough to provide weak anonymity.

3.4 Revokability

To avoid frauds mentioned in the introduction, induced by perfect anonymity, a possible revocation of anonymity has become a basic requirement to electronic-cash schemes. This means that, when the need arises (with fraud evidences) a third-party (the Revocation Center) can recover the link between a payment and a withdrawal (and therefore the user), and prove the validity of this link to anybody.

3.5 Diagram of Trust

About personal informations, nobody trusts nobody else for the use or abuse that can be made with them. For example, the bank could get profit from some relevant information about users: what he reads, where he buys bred, etc.

It is clear that the RC will have to be trusted, for the anonymity concerns, since he can trace any transaction. However, in case of fraud, his revealed informations should not be trusted by a judge, without any proof of validity, as he may want to protect someone.

All the other participants (the bank, the APs and the shop) cannot be trusted. Therefore, we want them (any group of them) not to be able to reveal and prove a link between a user and a transaction.

4 Intuition

With the model described above, one may attempt to informally present a new candidate to provide anonymity, we will call "self-scrambling anonymizer".

4.1 Withdrawal

As usual, a revocable-anonymous coin is a certified message, which embeds the user's public key. In our setting, the message is simply an encryption of this user's public key (pk_{U}), using the public key of the RC ($\mathsf{pk}_{\mathrm{RC}}$). Using the encryption of the user's identity as electronic coin has already been done by Camenisch *et al.* [8]. It is very convenient for anonymity revocation since the identity of the

The user/consumer	The bank
1. computes an encryption $c = E(\mathsf{pk}_{\mathrm{RC}}, \mathsf{pk}_{\mathrm{U}}; r)$ of his public key pk_{U} with the random r, 2. produces a signature $\sigma = S(\mathsf{sk}_{\mathrm{U}}, (\mathsf{pk}_{\mathrm{U}}, r, c))$ on it 3. sends the triple $(\mathsf{pk}_{\mathrm{U}}, r, \sigma)$ to the bank.	1. recovers $c = E(\mathsf{pk}_{\mathrm{RC}}, \mathsf{pk}_{\mathrm{U}}; r)$, 2. checks the signature σ 3. returns a certificate Cert_c on c. The coin consists of the pair (c, Cert_c).

Fig. 1. Withdrawal

owner of a coin involved in a fraudulent transaction can be easily recovered by the RC, using its secret key.

But instead of using intricate zero-knowledge proofs to convince the bank of the validity of the encryption, the user shows everything to the bank (the public key and the random coins used for the encryption, see Figure 1), and even signs it. So that the bank certifies the encryption with full confidence. Then, the resulting coin will be used without any further modification, such as heavy (restrictive) blinding processes.

4.2 Anonymity Process

But then, where is anonymity? Indeed, the bank knows the coin and can easily trace any transaction performed through its use, and convince anyone of the validity of this information, by providing the construction of the ciphertext. Then, appear the "Anonymity Providers" who will help the user to make this coin anonymous: the user can derive a new encryption c' of his identity (thus "self-scrambling") in an indistinguishable way. However, since he gets a new ciphertext c', he needs a new certificate. An AP can provide this new certificate. But before certifying c' he requires both the previous coin (c, Cert_c) and the proof of equivalence between the two ciphertexts.

1. From the old coin (c, Cert_c), the user derives a new encryption of his identity c'.
2. He provides both the old coin and the proof that c and c' encrypt the same public key.
3. Then he receives a certificate $\mathsf{Cert}_{c'}$ on c', from the AP.

Fig. 2. Anonymity Process (first sketch)

4.3 Security, Anonymity and Revokability

By now, the greatest problems appear: how can one be ensured anonymity, without any risk of fraud?

– On the one hand, one wants to avoid traceability of coins, and at least achieve weak anonymity. To address this problem, we use a proof technique that just convinces the involved AP, in a "non-transferable" way. Thanks to this "designated verifier" proof, this latter is unable to convince anyone else with the resulting transcript of the proof.

– On the other hand, the user owns two coins which represent the same money (the old and the new coins), but can exchange or spend both of them, which results in an over-spending! To cover himself, the AP needs to be able to prove that he gave a coin for another coin, so that the fraud might only have been performed by the user. To allow that, the previous "designated verifier" proof must furthermore be "undeniable" by the owner of the coin: if the AP asks the user to confirm the transcript he holds, the user cannot deny. However, if the AP has produced by himself a wrong transcript, the user will be able to deny it.

As a consequence, the proof of equivalence between the two ciphertexts is done using a "designated verifier undeniable signature" [28] which first just convinces the AP, in a non-transferable way. But the user won't be able to deny later this transcript.

4.4 Anonymity Provider: Self-Scrambling Anonymizer

With the above definitions, one can outline the process of a "self-scrambling anonymizer" (see Figure 3). We just assume that the user owns a valid coin $c = \mathcal{E}(\mathsf{pk}_{\text{RC}}, \mathsf{pk}_{\text{U}}; y)$ with its certificate Cert_c, which guarantees correct withdrawal from the bank, and therefore a possible revocation. At the end of the process, he owns a new valid coin, $c' = \mathcal{E}(\mathsf{pk}_{\text{RC}}, \mathsf{pk}_{\text{U}}; y + t)$ with its certificate $\mathsf{Cert}_{c'}$.

1. The user re-encrypts the coin c into $c' = \mathcal{E}(\mathsf{pk}_{\text{RC}}, \mathsf{pk}_{\text{U}}; y + t)$
2. The user provides an undeniable signature s, using c as a public key associated with the secret key $(\mathsf{sk}_{\text{U}}, y)$, of the equivalence between c and c'. This latter equivalence is guaranteed by the existence of t.
3. The user confirms the validity of this signature s to the AP (and only him).
4. The AP certifies the new coin c' and sends $\mathsf{Cert}_{c'}$ to the user.

Fig. 3. Self-Scrambling Anonymizer

This validity of this process is quite obvious. Indeed, after steps 2 and 3, the AP is convinced of that

– the conversion has been performed by the owner of the coin c;
– c' is equivalent to c.
– the owner of c won't be able to deny later s (the relation between c and c');

4.5 Spending and Deposit

When a user possesses a coin (anonymous or not), he can simply spend it by proving he really owns it: he proves his knowledge of the secret key $(\mathsf{sk}_{\text{U}}, y)$ associated to the public key $c = \mathcal{E}(\mathsf{pk}_{\text{RC}}, \mathsf{pk}_{\text{U}}; y)$. This proof is a signature related to the purchase which will also convince the bank when the shop deposits the coin.

5 A Practical Example

To illustrate the efficiency and even practicability of such a "self-scrambling anonymizer", we give an example where anonymity is based on the decisional Diffie-Hellman problem [5].

In what follows, H always denotes a hash function, assumed to behave like an ideal random function [4]. We begin with the structure of a coin, based on the El Gamal's encryption. Then, we describe how one can prove his ownership of a coin. Finally, we describe an efficient undeniable proof of equivalence of coins. In the discrete logarithm setting, many such proofs has been proposed. But since we want an optimistic-oriented scheme, we focus on an efficient protocol for the confirmation. Indeed, the disavowal process is only needed in case of dispute.

5.1 El Gamal Encryption

As we have already seen, the El Gamal's encryption scheme [18] is a public key encryption scheme which meets the semantic security. Let us briefly recall it, see on Figure 4.

- The system needs a group $\mathcal{G}$ of order q, and a generator g.
  The secret key is an element $X \in \mathbb{Z}_q$ and the public key is $Y = g^X$.
- For any message $m \in \mathcal{G}$, $c = \mathcal{E}(Y, m; r) = (g^r, Y^r m)$, for $r \stackrel{R}{\leftarrow} \mathbb{Z}_q^\star$.
- For any ciphertext $c = (a, b)$, $m = \mathcal{D}(X, c) = b/a^X$.

Fig. 4. El Gamal Encryption Scheme

5.2 Proof of Ownership of a Coin

Let us assume that Y is the public key of the Revocation Center, and $I = g^x$ the identity of a user. A coin is an encryption of I: $c = (a = g^r, b = Y^r I)$ which is afterwards certified by the bank. With the certificate of the bank, one knows that the encryption is valid. Therefore, in order to prove his ownership, the user has just to convince of his knowledge of (x, r) such that $b = Y^r g^x$. This can be done using another signature scheme proposed by Okamoto [33] and recalled on Figure 5.

- The prover chooses random $k, s \in \mathbb{Z}_q$ and computes $t = Y^k g^s$
- he produces a challenge, depending on the message m: $e = H(m, t)$
- he then computes $u = k - re \bmod q$ and $v = s - xe \bmod q$.
- The signature finally consists of the triple (e, u, v).
- In order to verify it, one has just to compute $t' = Y^u g^v b^e$
  and check whether $e = H(m, t')$ or not.

Fig. 5. Proof of Validity of a coin $c = Y^r g^x$

Then, a scrambled coin is simply got by multiplying both parts of the old one by respective bases, g and Y, put at a same random exponent ρ:

$$c' = (a' = g^\rho a, b' = Y^\rho b) = (g^{r+\rho}, Y^{r+\rho} I).$$

Then, if the owner of the old coin has certified (relative to the public key c) the message $m = h^\rho$, equivalence of both coins can be proven with the proof of equivalence of three discrete logarithms

$$\log_h m = \log_g(a'/a) = \log_Y(b'/b).$$

This can be efficiently done, interactively, using the protocol presented below on Figure 6, or in a non-interactive way, using the protocol presented on Figure 7, where the value $y_V = g^{x_V}$ is the public key of the designated verifier, the AP.

5.3 Proof of Equality of Many Discrete Logarithms

Let us review the classical protocol used to prove the equality of many discrete logarithms in a zero-knowledge way. A group $\mathcal{G}$ is given with $k+1$ independent generators g, h_1, ..., h_k (which means that nobody knows the relative discrete logarithms) of order q.

The prover owns a pair $(x, y = g^x)$, and wants to prove that for some z_i in $\mathcal{G}$, for $i = 1, \ldots, k$, $z_i = h_i^x$ (with the same x as above) without revealing x. This can be done using the interactive zero-knowledge protocol presented on Figure 6, which is clearly designated-verifier, as any interactive zero-knowledge proof, thanks to the simulatability of the transcript. We insist on the fact that the interactive protocol needs to be zero-knowledge in the strong sense, not only against honest verifiers: the challenge must be of fixed short length while the protocol has to be iterated to reach a level of security.

1. the prover chooses a random value $u \in \mathbb{Z}_q$,
   computes $b = g^u$ and $c_i = h_i^u$
   and sends $d = H(b, c_1, \ldots, c_k)$ to the prover.
2. the verifier chooses a random challenge $e \in \{0, \ldots, 2^\ell\}$ and sends it to the prover.
3. the prover computes $f = u - xe \bmod q$
   and sends f to the verifier.
4. the verifier checks that
   $$d = H(g^f y^e, h_1^f z_1^e, \ldots, h_k^f z_k^e).$$

Fig. 6. Zero-Knowledge Proof of Equality of Discrete Logarithms

On figure 7, is presented the non-interactive version which is turned into a designated-verifier signature thanks to the trapdoor-commitment [28] which can be opened in any way by the verifier who knows the discrete logarithm x_V of y_V relatively to g.

This provides a designated verifier non-interactive proof. Indeed, the verifier is convinced of the equality of many discrete logarithms, but since he could have opened the commitment a in any way he wanted, thanks to the knowledge of the discrete logarithm of y_V in the basis g, such a proof cannot convince anyone else.

1. the prover chooses random values $u, v, w \in \mathbb{Z}_q$
   and computes $a = g^w y_V^v$
   $b = g^u$ and $c_i = h_i^u$
   $e = H(a, b, c_1, \ldots, c_k)$
   $f = u - x(e + w) \bmod q$
   the signature consists of the tuple (e, f, w, v).
2. the verifier checks that

$$e = H(g^w y_V^v, g^f y^{e+w}, h_1^f z_1^{e+w}, \ldots, h_k^f z_k^{e+w}).$$

Fig. 7. Designated-Verifier NIZK Proof of Equality of Discrete Logarithms

6 A Complete Description: an Electronic Cash Scheme

6.1 Description

This candidate provides a very simple electronic cash scheme with revocable anonymity. Let us assume g and h to be independent elements in $\mathcal{G}$ of order q. The RC public key is $Y = g^X$.

1. **Registration**: The user chooses a secret key $x \in \mathbb{Z}_q$ and publishes $I = g^x$, which is then certified by a Certification Authority, Cert_I, after verification of ID card or driving license.
2. **Withdrawal**: The user $I = g^x$ constructs a coin $c = (a = g^r, b = Y^r I)$, using the public key Y of the Revocation Center. He also signs c together with the date, using his private key x and a Schnorr signature [40]. He sends both to the bank together with r, I and Cert_I. Then the bank can check both the validity of I (with the signature of the coin and the date, only the legitimate user could have done it) and the correct encryption, so that the RC can revoke anonymity at anytime. After having modified the user's account, the bank sends back a certificate Cert_c. The user just has to remember r and Cert_c, which is just 50-bytes long (or even less if r is pseudo-randomly generated).
3. **Self-Scrambling Anonymizer**: To get some anonymity, the user contacts any AP.
   - The user chooses a random ρ and "self-scrambles" the coin:

$$c' = (a' = g^\rho a, b' = Y^\rho b).$$

   - He produces a signature $S = (e, u, v) = \mathcal{S}((r, x), m)$ on $m = h^\rho$ using the secret key (r, x) related to the public one $b = Y^r g^x$ (extracted from the coin c) as shown on Figure 5.

     Remark 7. Because of S, the user (the owner of c) won't be able to deny later his knowledge of ρ. Furthermore, nobody can impersonate the user at this step, even the RC, since the discrete logarithm x of I is required to produce a valid signature (no existential forgery).

   - He also provides a designated-verifier proof (using either the simple interactive zero-knowledge proof or the non-interactive one using the AP's public key y_V, as shown on Figure 7) of equality of discrete logarithms $DVP = (e, f, w, v)$,

$$\log_h m = \log_g(a'/a) = \log_Y(b'/b).$$

- He finally sends $c = (a, b), \mathsf{Cert}_c, c' = (a', b'), m, S$ and DVP to the AP.
- The AP checks the certificate Cert_c on c, the validity of the signature S on the message m using the public key b (which consists in the test $e = H(m, Y^u g^v b^e)$), and the validity of DVP:

$$H(g^w y_V^v, h^f m^{e+w}, g^f (a'/a)^{e+w}, Y^f (b'/b)^{e+w}) \stackrel{?}{=} e \bmod q.$$

  He then certifies c' and sends back this certificate, $\mathsf{Cert}_{c'}$, to the user.

Remark 8. After this just 2-round process the user gets a new certified coin $\{c' = (g^{r+\rho}, Y^{r+\rho} I), \mathsf{Cert}_{c'}\}$ which is now strongly anonymous from the point of view of the bank (or any previous AP). The user can then erase the old coin and put $(r+\rho, \mathsf{Cert}_{c'})$ instead, keeping also ρ somewhere since he has certified his knowledge of it. Therefore, his space requirement just increases by 20 bytes at each anonymity step (or less if ρ is pseudo-randomly generated). On the other hand, AP has to keep (c, c', m, S) to be able to prove the link between c and c', with the help of the user. Whereas DVP cannot help him to convince anyone.

4. **Spending**: When the user wants to spend a coin, he just gives it together with a signature $S = (e, u, v)$ of the purchase, date, etc, with the secret key associated to the coin (which proves the ownership of the coin) to the payee.
5. **Revocation**: If a coin is used twice or more (spent or made anonymous), which can be proven by showing two different signatures S and S' involving this coin, identity I can easily be recovered by the RC, simply decrypting the coin c.

   If the user refutes the revealed identity, the RC can prove the value of the identity embedded in the considered coin. Since the owner of the coin (the guy who's identity is embedded in the coin) has been able to produce the signature, this proves the identity of the bad guy.

6.2 Security Concerns

Anonymity. First, one can see that weak anonymity is obtained after the use of just one AP. Indeed, this AP knows the relation between the two coins (and only him), but he cannot prove it. The proof he got was just for him and cannot convince anyone else.

However, he "knows" the link and can reveal it. And this may annoy some people who would like strong anonymity. Therefore, they can make use of many other APs. Just one honest AP (who does not reveal the links he knows) is enough for strong anonymity. Indeed, all the APs and the bank must cooperate to trace a transaction.

Therefore, a high level of anonymity can be obtained with few APs. However, for efficiency concern, if some transactions do not require such an anonymity, the user can directly spend the coin obtained from the bank.

Impersonation. As already seen, the secret key x of a user is never revealed, but only used in some signatures or zero-knowledge proofs. Any user is therefore protected against any impersonation, even from a collusion of the bank, the APs and even the RC, since this secret key is required in any transaction, to certify the ownership.

Forgery. Because of the security of the signature used to certify a coin, counterfeit money is infeasible for any user.

However, any AP has the ability to create money. To avoid such a forgery from the APs, they can be seen as middlemen: an AP sends a new coin c' against another coin c. And then, he asks money for c to the provider of c (the provider of c does the same thing, and so on, up to the bank).

If an AP cooperates with a user and certifies more coins than he receives, he will be asked for more money than he received. He will pay for the user, since he won't be able to show the original coin linked to this suspected one, with the undeniable signature from the user.

Fraud Detection. Thanks to this structure using APs as middlemen, an over-spending can be easily detected: if a user tries to anonymize one coin c twice (to obtain two new ones), the provider of c will be asked money twice for the same coin c. The fraud will be detected and proved with two signatures from the user.

Thanks to the undeniable signature, the successive coins, anonymously generated from the fraudulent ones, can be traced.

Privacy Revocation. As usual a revocable anonymous e-cash scheme requires

- Payment-based tracing: upon over-spending, proven by many uses of a same coin, the RC can recover (and prove) the identity I of the fraudulent guy.
- Withdrawal-based tracing: if a user has been forced to give some coins, to a criminal, he just reveals these coins, which will be blacklisted. If some secret information has been stolen (for example the secret key), this information can be made public to help anybody to refuse any transaction performed with this secret.

6.3 Improvements

This scheme admits many variations to improve both efficiency and security, and then to make it more realistic.

Security. Let us first consider the security. To enhance it, one can use a distributed RC which runs a threshold cryptosystem [16]. Then, anonymity won't be revoked with a one-man's decision.

Efficiency. This scheme is already very efficient, since each "self-scrambling anonymizer" phase only requires 10 exponentiations from the user point of view and 11 from the AP's point of view.

If we consider any AP as a middleman which gives a new coin for an old one, he can also gives many smaller new ones for an old one, which is slightly more efficient than getting many small coins from the bank and asking to the AP to anonymize each of them.

Profitability. To make such a frame realistic, the AP business must be profitable: for example, he can give back coins of just 99.9% the value of the old one, and keeps the rest as profit.

Therefore, a user is charged for anonymity. The more anonymity he wants, the more he is charged. The profitability of this business makes it realistic: anonymity has a price which has to be paid just by people who want it (and not by the banks which do not really need/want it).

7 Conclusion

In this paper, we have proposed a new tool to provide revocable and scalable anonymity at no risk for the user. The scalability of this scheme makes it quite efficient: just the user who wants anonymity has to pay the computational cost. Furthermore, this "Self-Scrambling Anonymity" process can be performed with the help of an "Anonymity Provider" who can also financially charge the user. This may become a new profitable business.

Moreover, we hope that our "self-scrambling anonymizer" tool may have other applications, because of its flexibility and efficiency, anywhere anonymity is required.

8 Acknowledgments

I wish to thank Markus Jakobsson for many valuable comments.

References

1. M. Abe. Universally Verifiable Mix-Net with Verification Work Independent of the Number of Mix-Servers. In *Eurocrypt '98*, LNCS 1403, pages 437–447. Springer-Verlag, Berlin, 1998.
2. M. Abe. Mix-Networks on Permutation Networks. In *Asiacrypt '99*, LNCS 1716. Springer-Verlag, Berlin, 1999.
3. M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations among Notions of Security for Public-Key Encryption Schemes. In *Crypto '98*, LNCS 1462, pages 26–45. Springer-Verlag, Berlin, 1998.
4. M. Bellare and P. Rogaway. Random Oracles Are Practical: a Paradigm for Designing Efficient Protocols. In *Proc. of the 1st CCS*, pages 62–73. ACM Press, New York, 1993.
5. S. A. Brands. An Efficient Off-Line Electronic Cash System Based on the Representation Problem. Technical Report CS-R9323, CWI, Amsterdam, 1993.
6. S. A. Brands. Restrictive Blinding of Secret-Key Certificates. Technical Report CS-R9509, CWI, Amsterdam, 1995.
7. E. Brickell, P. Gemmell, and D. Kravitz. Trustee-based Tracing Extensions to Anonymous Cash and Making of Anonymous Change. In *SODA '95*, pages 457–466, 1995.
8. J. Camenisch, J.-M. Piveteau, and M. Stadler. Fair Blind Signatures. In *Eurocrypt '95*, LNCS 921, pages 209–219. Springer-Verlag, Berlin, 1995.
9. J. Camenisch, J.-M. Piveteau, and M. Stadler. An Efficient Fair Payment System. In *Proc. of the 3rd CCS*, pages 88–94. ACM Press, New York, 1996.

10. D. Chaum. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Communications of the ACM*, 24(2):84–88, February 1981.
11. D. Chaum. Blind Signatures for Untraceable Payments. In *Crypto '82*, pages 199–203. Plenum, New York, 1983.
12. D. Chaum. Zero-Knowledge Undeniable Signatures. In *Eurocrypt '90*, LNCS 473, pages 458–464. Springer-Verlag, Berlin, 1991.
13. D. Chaum, A. Fiat, and M. Naor. Untraceable Electronic Cash. In *Crypto '88*, LNCS 403, pages 319–327. Springer-Verlag, Berlin, 1989.
14. D. Chaum and H. van Antwerpen. Undeniable Signatures. In *Crypto '89*, LNCS 435, pages 212–216. Springer-Verlag, Berlin, 1990.
15. D. Chaum, E. van Heijst, and B. Pfitzmann. Cryptographically Strong Undeniable Signatures, Unconditionally Secure for the Signer. In *Crypto '91*, LNCS 576, pages 470–484. Springer-Verlag, Berlin, 1992.
16. Y. Desmedt and Y. Frankel. Threshold Cryptosystems. In *Crypto '87*, LNCS 293, pages 307–315. Springer-Verlag, Berlin, 1988.
17. W. Diffie and M. E. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, IT–22(6):644–654, November 1976.
18. T. El Gamal. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. *IEEE Transactions on Information Theory*, IT–31(4):469–472, July 1985.
19. A. Fiat and A. Shamir. How to Prove Yourself: Practical Solutions of Identification and Signature Problems. In *Crypto '86*, LNCS 263, pages 186–194. Springer-Verlag, Berlin, 1987.
20. E. Fujisaki and T. Okamoto. Practical Escrow Cash Systems. In *Security Protocols*, 1996.
21. S. Goldwasser and S. Micali. Probabilistic Encryption. *Journal of Computer and System Sciences*, 28:270–299, 1984.
22. S. Goldwasser, S. Micali, and R. Rivest. A Digital Signature Scheme Secure Against Adaptative Chosen-Message Attacks. *SIAM Journal of Computing*, 17(2):281–308, April 1988.
23. M. Jakobsson. Blackmailing using Undeniable Signatures. In *Eurocrypt '94*, LNCS 950, pages 425–427. Springer-Verlag, Berlin, 1995.
24. M. Jakobsson. A Practical Mix. In *Eurocrypt '98*, LNCS 1403, pages 448–461. Springer-Verlag, Berlin, 1998.
25. M. Jakobsson. Flash Mixing. In *Proc. of the 18th PODC*, pages 83–89. ACM Press, New York, 1999.
26. M. Jakobsson and D. M'Raïhi. Mix-based Electronic Payment. In *Proc. of the Fifth Annual Workshop on Selected Areas in Cryptography*, LNCS 1556, pages 157–173. Springer-Verlag, Berlin, 1998.
27. M. Jakobsson and J. Müller. Improved Magic Ink Signatures Using Hints. In *Financial Cryptography '99*, LNCS 1648, pages 253–268. Springer-Verlag, Berlin, 1999.
28. M. Jakobsson, K. Sako, and R. Impagliazzo. Designated Verifier Proofs and Their Applications. In *Eurocrypt '96*, LNCS 1070, pages 143–154. Springer-Verlag, Berlin, 1996.
29. M. Jakobsson and C. P. Schnorr. Efficient Oblivious Proofs of Correct Exponentiation. In B. Preneel, editor, *Proc. of CMS '99*. Kluwer Academic Publishers, Boston, 1999.
30. M. Jakobsson and M. Yung. Distributed "Magic Ink" Signatures. In *Eurocrypt '97*, LNCS 1233, pages 450–464. Springer-Verlag, Berlin, 1997.
31. A. Juels, M. Luby, and R. Ostrovsky. Security of Blind Digital Signatures. In *Crypto '97*, LNCS 1294, pages 150–164. Springer-Verlag, Berlin, 1997.
32. M. Michels and M. Stadler. Efficient Convertible Undeniable Signature Schemes. *Fourth Annual Workshop on Selected Areas in Cryptography* available from http://www.scs.carleton.ca/~sac97, 1997.
33. T. Okamoto. Provably Secure and Practical Identification Schemes and Corresponding Signature Schemes. In *Crypto '92*, LNCS 740, pages 31–53. Springer-Verlag, Berlin, 1992.
34. D. Pointcheval and J. Stern. Security Proofs for Signature Schemes. In *Eurocrypt '96*, LNCS 1070, pages 387–398. Springer-Verlag, Berlin, 1996.
35. D. Pointcheval and J. Stern. Security Arguments for Digital Signatures and Blind Signatures. *Journal of Cryptology*, 2000.
Available from `http://www.di.ens.fr/~pointche`.
36. M. O. Rabin. Digitalized Signatures. In R. Lipton and R. De Millo, editors, *Foundations of S ecure Computation*, pages 155–166. Academic Press, New York, 1978.
37. C. Radu, R. Govaerts, and J. Vanderwalle. A Restrictive Blind Signature Scheme with Applications to Electronic Cash. In *Communications and Multimedia Security II*, pages 196–207. Chapman & Hall, London, 1996.

38. C. Radu, R. Govaerts, and J. Vanderwalle. Efficient Electronic Cash with Restricted Privacy. In *Financial Cryptography '97*, LNCS. Springer-Verlag, Berlin, 1997.
39. M. K. Reiter and A. D. Rubin. Crowds, anonymous web transactions. *ACM Transactions on Information and System Security*, 1:66–92, 1998.
40. C. P. Schnorr. Efficient Signature Generation by Smart Cards. *Journal of Cryptology*, 4(3):161–174, 1991.
41. S. von Solms and D. Naccache. On Blind Signatures and Perfect Crimes. *Computers & Security*, 11:581–583, 1992.