Coin-Based Anonymous Fingerprinting

Birgit Pfitzmann, Ahmad-Reza Sadeghi

Universität des Saarlandes, Fachbereich Informatik,
D-66123 Saarbrücken, Germany
Email: {pfitzmann, sadeghi}@cs.uni-sb.de

Abstract. Fingerprinting schemes are technical means to discourage people from illegally redistributing the digital data they have legally purchased. These schemes enable the original merchant to identify the original buyer of the digital data. In so-called asymmetric fingerprinting schemes the fingerprinted data item is only known to the buyer after a sale and if the merchant finds an illegally redistributed copy, he obtains a proof convincing a third party whom this copy belonged to. All these fingerprinting schemes require the buyers to identify themselves just for the purpose of fingerprinting and thus offer the buyers no privacy. Hence anonymous asymmetric fingerprinting schemes were introduced, which preserve the anonymity of the buyers as long as they do not redistribute the data item.
In this paper a new anonymous fingerprinting scheme based on the principles of digital coins is introduced. The construction replaces the general zero-knowledge techniques from the known certificate-based construction by explicit protocols, thus bringing anonymous fingerprinting far nearer to practicality.

1 Introduction

Fingerprinting schemes are cryptographic techniques supporting the copyright protection of digital data. They do not require tamper-resistant hardware, i.e., they do not belong to the class of copyright protection methods which *prevent* copying. It is rather assumed that buyers obtain data in digital form and can copy them. Buyers who redistribute copies disregarding the copyright conditions are called *traitors*. Fingerprinting schemes discourage traitors by enabling the original merchant to identify a traitor who originally purchased the data item. Every sold copy is slightly different from the original data item and unique to its buyer. Obviously the differences to the original represent the information embedded in the data item, which must be imperceptible. As several traitors might collude and compare their copies to find and eliminate the differences, cryptographic methods were used to make fingerprinting schemes *collusion tolerant*. There are different classes of fingerprinting schemes called *symmetric* [BMP86,BS95] and *asymmetric* [PS96,PW97a,BM97]. In contrast to symmetric schemes, asymmetric schemes require the data item to be fingerprinted via an interactive protocol between the buyer and the merchant where the buyer also inputs her own secret. At the end of this protocol only the buyer knows the fingerprinted data item. However, after finding a redistributed copy the merchant can extract information which not only enables him to identify a traitor but also provides him with a proof of treachery that convinces any third party. The main construction in [PS96] was based on general primitives; an *explicit* construction, i.e., without such primitives, was only given for the case without significant collusions. Explicit collusion-tolerant constructions were given in [PW97a,BM97]. A special variant of fingerprinting is traitor tracing [CFN94,NP98]; here the keys for broadcast encryption [FN94] are fingerprinted. Asymmetric traitor tracing was introduced in [Pfi96] with a construction based on general primitives. Explicit constructions for this case were also given in [PW97a]. Even more efficient construction were given in [KD98]; however, they are not asymmetric in the usual sense but "arbitrated", i.e., a certain number of predefined arbiters can be convinced by the merchant (similar to the difference between arbitrated authentication codes and asymmetric signature schemes).

As in "real-life" market places, it is desired that electronic market places offer privacy to the customers. It should be possible to buy different articles (pictures, books, etc.) anonymously, since buying items can reveal a lot of behavioristic information about an individual. To allow this also for buying fingerprinted items, *anonymous asymmetric fingerprinting schemes* were proposed [PW97b]. Note that in normal fingerprinting (symmetric and asymmetric) the buyer has to identify herself during each purchase. In anonymous fingerprinting the anonymity of the buyers is preserved as long as they do not redistribute copies of the data item.

In this paper, we introduce a new anonymous asymmetric fingerprinting scheme based on the principles of digital coins. Our protocols are explicit, in contrast to the scheme in [PW97b], where general theorems like "every NP-language has a zero-knowledge proof system" were used, and thus far more efficient. The anonymity and security are computational. Security of one party relies on a restrictiveness assumption about the underlying payment system, which we formulate precisely.

2 The Model of Anonymous Fingerprinting

The involved parties in the model are merchants $\mathcal{M}$, buyers $\mathcal{B}$, registration centers $\mathcal{RC}$ and arbiters $\mathcal{A}$. For the purpose of fingerprinting it is required in this model that buyers register themselves to a registration center $\mathcal{RC}$ (e.g., their bank). The required trust in $\mathcal{RC}$ should be minimum such that a cheating $\mathcal{RC}$ can only refuse a registration.[1] It is assumed that $\mathcal{B}$ can generate signatures (using an arbitrary signature scheme) under her "real" identity $ID_{\mathcal{B}}$ and that the corresponding public keys have already been distributed. Furthermore there is no special restriction on the arbiter $\mathcal{A}$. Any third party having access to the corresponding public keys should be convinced by the proof. The main subprotocols of the construction are registration, fingerprinting, identification, and trial. Identification includes a variant "enforced identification" for the case where $\mathcal{RC}$ refuses to cooperate. The main security properties are:

- Security for the merchant: As long as collusions do not exceed a certain size, the merchant will be able to identify a traitor for each illegally redistributed item and to convince any honest arbiter. (In case $\mathcal{RC}$ colludes with the traitors the identified traitor may be $\mathcal{RC}$.)

[1] In particular, even a collusion of $\mathcal{M}$ and $\mathcal{RC}$ should not be able to trace a buyer who did not redistribute a data item. Otherwise one can trivially use any known non-anonymous asymmetric scheme and simply let the initial key pair of the buyer be certified under a pseudonym whose owner is only known to the certification authority. This was overlooked in [Dom98].

– Security for the buyer and $\mathcal{RC}$: Nobody is unduly identified as a traitor; at least no honest arbiter will believe it.
– Anonymity as sketched above; different purchases by one buyer should also be unlinkable.

For the detailed definitions of the subprotocols and security properties we refer the interested reader to [PW97b].

3 Overview of the Construction

To see more precisely what we achieve, note that [PW97b] contains a modular construction: The first part, called framework, is a construction based on certificates. At the end, the merchant holds a commitment com to a certain value emb and possibly other information. The buyer can open the commitment and may also hold other information. This framework guarantees that whenever the merchant later obtains emb, he can identify this buyer and win a trial against him. In the second part, the value emb is embedded into the data item in a way that does not release additional information about emb. The embedding procedure must guarantee that whenever a collusion of at most the maximum tolerated size redistributes a data item, the merchant will be able to reconstruct the value emb that was used by at least one traitor. For the second part, constructions were previously only known for the case of traitor tracing and for normal fingerprinting without collusion tolerance. The latter is explicit, and one can see quite easily that the former (based on [PW97a] Section 4) can also be made explicit by using the efficient key selection protocol from Section 2.3 of the same paper in the appropriate places. The main technical part in [PW97b] was to construct a suitable collusion-tolerant embedding procedure for normal fingerprinting.

In contrast, the framework part in [PW97b] is a comparatively simple construction where emb, the content of the commitment, is a signature with respect to a key that the merchant must not know, and the buyer proves with a general zero-knowledge technique that she knows such a key and a certificate by $\mathcal{RC}$ on it. It is this first part that we replace with the explicit and much more efficient coin-based construction. It can then be combined with the known second parts. This gives us explicit overall constructions for collusion-tolerant anonymous traitor tracing and for anonymous normal fingerprinting without collusion tolerance. For collusion-tolerant normal fingerprinting, the construction in [PW97b] is not explicit, but it is mentioned that all the steps in a secure 2-party computation look quite simple so that it should be possible to find simpler explicit realizations for them. This can in fact be done (each time exploiting either homomorphism – note that even the Reed-Solomon codes are a linear operation – or the efficient table-lookup from [PW97b], end of Section 2.3), but we do not attempt to include details of that here.

The basic idea for using digital cash systems with double-spender identification to construct an anonymous fingerprinting scheme is as follows: Registration will correspond to withdrawing a coin. (The "coins" serve only as a cryptographic primitive and have no monetary value.) The untraceability of the cash system will give us the unlinkability of the views of the registration center and the merchant. Redistribution of a data item should correspond to double-spending of the underlying coin, i.e., the value emb

embedded in the data item will be similar to the response in a second payment with the coin. We could execute a complete first payment during fingerprinting, but actually our protocols can be simpler. (They are more like "zero-spendable" coins where each coin as such can be shown, but any response to a challenge leads to identification.) One new problem is that while in a payment the response is given to the merchant in clear, in our case it must be verifiably hidden in a commitment. Another problem is that double-spender identification is usually a binary decision. In our case, however, it must be decided reliably under what copyright conditions the redistributed item was bought – e.g., there may be items that can be redistributed after a while. In the formal security requirements this is a value $text$ input in fingerprinting and also in a trial. Thus the identification information must be linked to such a text during fingerprinting in a way that even a collusion of a merchant and the registration center cannot forge, although such a collusion can sign additional coins that look like belonging to a specific buyer.

4 Construction

Our explicit construction employs the ideas from the digital cash scheme in [Bra94], or, as we do not have the same double-spender identification, at least from the underlying blind signature scheme [CP93]. We make the following conventions:

Algebraic Structure: All arithmetic operations are performed in a group G_q of order q for which efficient algorithms are known to multiply, invert, determine equality of elements, test membership and randomly select elements. Any group G_q satisfying these requirements and in which the computation of discrete logarithms is infeasible can be a candidate. For concrete constructions one can assume that G_q is the unique subgroup of prime order q of the multiplicative group $\mathbb{Z}_p^*$ where p is a prime such that $q|(p-1)$.

Hash Function: Hash functions are denoted by $hash$. We have to make the same assumptions as in [Bra94], which is behavior similar to a random oracle.

Commitment Scheme: We use two commitment schemes. The first one is based on discrete logarithms in G_q (see [BKK90] for the one-bit version and [CHP92,Ped92] based on [BCP88] for the general version). To commit to a value $b \in \mathbb{Z}_q$, the committer needs generators $g'', h'' \in_R G_q \setminus \{1\}$, which are typically randomly generated and sent by the recipient. Then the committer selects $x \in_R \mathbb{Z}_q$ and computes the commitment $y = BC^{DL}(x, b) = g''^b h''^x \mod p$. To open y, the committer reveals (b, x). This scheme is information-theoretically hiding and it is binding under the discrete logarithm assumption in the corresponding group. Moreover due to the homomorphic property of this scheme one can commit to a number $r \in \mathbb{Z}_q^*$ using commitments to the bits of r: Let $r = \sum_{j=0}^{l-1} r_j 2^j$ be the binary representation of r and $BC^{DL}(x_j, r_j)$ be commitments to the bits. Then $\prod_{j=0}^{l-1}(BC^{DL}(x_j, r_j))^{2^j} = g''^r h''^x = BC^{DL}(x, r) \bmod p$ with $x = \sum_{j=0}^{l-1} x_j 2^j \bmod q$.

The second bit commitment scheme is based on quadratic residues (see [GM84,BCC88]). To commit to a bit b' the committer computes $y = BC^{QR}(x', b') = (-1)^{b'} x'^2 \mod n$ where $x' \in_R \mathbb{Z}_n^*$ and n is a Blum integer chosen by the committer.

4.1 Key Distribution and Registration

Registration Center Key Distribution: $\mathcal{RC}$ randomly selects a group G_q and generators g, g_1, $g_2 \in_R G_q \setminus \{1\}$ and a number $x \in_R \mathbb{Z}_q^*$ as its secret key. $\mathcal{RC}$ also chooses a hash function $hash$. It publishes the group description, (g, g_1, g_2), its public key $h \equiv g^x \bmod p$ and $hash$.

The correctness of the group description (i.e., that p and q primes with $q|(p-1)$ and generated in a way believed to exclude trap doors) and whether the generators are elements of $G_q \setminus \{1\}$ should be verified by other parties when using them. However, other parties do not rely on the randomness of the generators.

Opening a one-time account: This phase is similar to [Bra94], but in order to bind the identification information to a specific purchase text, each "account" is used only once. To open an account, $\mathcal{B}$ chooses $i \in_R \mathbb{Z}_q^*$ randomly and secretly and computes $h_1 \equiv g^i \bmod p$ (with $h_1 g_2 \neq 1$). She gives h_1 to $\mathcal{RC}$ and proves that she knows i (using the zero-knowledge proof from [CEG88] or, more efficiently, Schnorr identification [Sch91]). First $\mathcal{RC}$ verifies that the account number h_1 has not been used before. Then it stores h_1 in its registration database together with the claimed normal identity $ID_{\mathcal{B}}$ of this buyer. $\mathcal{B}$ gives a signature sig_{coin} on h_1 (with a suitable explanation) under her normal identity $ID_{\mathcal{B}}$ and $\mathcal{RC}$ verifies it. This signature can be used in later trials to show that $\mathcal{B}$ is responsible for this "account number" h_1.

Withdrawal: The protocol is shown in Figure 1. Essentially, $\mathcal{RC}$ signs the common input $m \equiv h_1 g_2 \equiv g_1^i g_2 \bmod p$ using a restrictive blind signature as in [Bra94]. Thus $\mathcal{B}$ obtains a signature $\sigma' = (z', a', b', r')$ on $m' \equiv m^s \equiv g_1^{is} g_2^s \bmod p$ where $s \in_R \mathbb{Z}_q^*$ is chosen randomly and secretly by $\mathcal{B}$.

Our protocol is also similar to the withdrawal protocol in [Bra94] in that an additional value is included in the hashing to obtain the challenge c' and thus in the signing process. In our case it is the public key of the key pair (sk_{text}, pk_{text}) from an arbitrary signature scheme, here Schnorr's for concreteness [Sch91]. We call the triple (m', pk_{text}, σ') a coin.

4.2 Fingerprinting

The fingerprinting subprotocol is executed between the (anonymous) buyer $\mathcal{B}$ and the merchant $\mathcal{M}$. This protocol differs (except for Step 1) from the payment protocol in [Bra94]. The common input is a text, $text$, describing the purchase item and licensing conditions.

Step 1: $\mathcal{B}$ selects an unused coin $coin' = (m', pk_{text}, \sigma')$. She uses the corresponding sk_{text} to make a Schnorr signature sig_{text} on $text$ and sends $(coin', sig_{text})$ to $\mathcal{M}$. Now $\mathcal{M}$ first verifies the validity of $coin'$ by computing $c' \equiv hash(m', z', a', b', pk_{text}) \bmod q$ and checking whether $m' \neq 1$ and $g^{r'} \equiv a' h^{c'}$ and $m'^{r'} \equiv b' z'^{c'} \bmod p$ hold [Bra94]. We say that a coin is valid if and only if it passes these tests. He then verifies sig_{text} using pk_{text} from $coin'$.

Step 2: $\mathcal{B}$ takes the internal structure (is, s) of $m' \equiv g_1^{is} g_2^s$ as the value to be embedded in the data item. Hence $emb = (is, s)$, and let $r_1 = is$, $r_2 = s$.

$$
\begin{array}{rcl}
\mathcal{B} & & \mathcal{RC} \\
 & & z \leftarrow m^x \\
 & & w \in_R \mathbb{Z}_q \\
 & & a \leftarrow g^w \bmod p \\
s \in_R \mathbb{Z}_q^* & \xleftarrow{\quad z,a,b \quad} & b \leftarrow m^w \bmod p \\
m' \leftarrow m^s,\ z' \leftarrow z^s & & \\
sk_{text} \in_R \mathbb{Z}_q^*,\ pk_{text} \leftarrow g_2^{sk_{text}} \bmod p & & \\
u,\ v \in_R \mathbb{Z}_q & & \\
a' \leftarrow a^u g^v,\ b' \leftarrow b^{su} m'^v & & \\
c' \leftarrow hash(m', z', a', b', pk_{text}) \bmod q & & \\
c \leftarrow c'/u \bmod q & \xrightarrow{\quad c \quad} & \\
g^r \stackrel{?}{\equiv} ah^c \bmod p,\ m^r \stackrel{?}{\equiv} bz^c \bmod p & \xleftarrow{\quad r \quad} & r \leftarrow cx + w \bmod q \\
r' \leftarrow ru + v \bmod q & &
\end{array}
$$

Fig. 1: The Withdrawal Part of the Registration Protocol

Since $\mathcal{M}$ should not get any useful information on this value, $\mathcal{B}$ hides it in a commitment. While the certificate-based framework in [PW97b] could leave the type of commitment open, we have to provide an efficient link from the given representation of emb, i.e., m', to the type of commitments needed in the later embedding procedures. For normal fingerprinting with and without collusion tolerance, these are quadratic residue commitments to individual bits of emb.[2]

We start this by producing discrete logarithm commitments to a binary representation of r_1 and r_2: The merchant $\mathcal{M}$ sends generators g'' and h'' to $\mathcal{B}$, and $\mathcal{B}$ sends back commitments $com_{kj} = g''^{r_{kj}} h''^{x_{kj}}$ where $r_k = \sum_{j=0}^{l-1} r_{kj} 2^j$ for $k = 1, 2$ and $x_{kj} \in_R \mathbb{Z}_q$. $\mathcal{M}$ may choose the generators randomly once for all its buyers. $\mathcal{B}$ should verify that they are elements of $G_q \setminus \{1\}$. From these individual commitments, $\mathcal{M}$ computes the commitments to r_1 and r_2, i.e., $com_k = \prod_{j=0}^{l-1} (com_{kj})^{2^j}$ for $k = 1, 2$. Now $\mathcal{B}$ proves the following predicate P to $\mathcal{M}$: The content of the commitments com_k is a pair $(r_1, r_2) \in \mathbb{Z}_q^* \times \mathbb{Z}_q^*$ with $m' \equiv g_1^{r_1} g_2^{r_2} \bmod p$. This can be done in zero-knowledge as shown in Figure 2, similar to other proofs concerning knowledge of representations of numbers with respect to certain generators following [CEG88]. As usual, we could also use larger challenges c at the cost of the real zero-knowledge property.

Note that this protocol does not prove that the values r_{kj} are binary; such a proof will be a side effect of Step 3.

[2] For traitor tracing, they are quadratic residue commitments to small blocks of emb represented in unary. Such a representation can also be derived efficiently from commitments to the bits.

$$
\begin{array}{rcl}
\mathcal{B} & & \mathcal{M} \\
& k = 1, 2 \text{ and } 0 \leq j \leq l-1 & \\
(r_1, r_2) \leftarrow emb = (is, s) & & \\
r_k = \sum_{j=0}^{l-1} r_{kj} 2^j & & \\
x_{kj} \in_R \mathbb{Z}_q,\ x_k \leftarrow \sum_{j=0}^{l-1} x_{kj} 2^j & & \\
com_{kj} \leftarrow g''^{r_{kj}} h''^{x_{kj}} & \xrightarrow{\quad com_{kj} \quad} & com_k \leftarrow \prod_{j=0}^{l-1} com_{kj}^{2^j} \\
\text{REPEAT } N \text{ times} & & \\
e_k, y_k \in_R \mathbb{Z}_q^* & & \\
V \leftarrow g_1^{e_1} g_2^{e_2} \bmod p & & \\
rcom_k \leftarrow g''^{e_k} h''^{y_k} & \xrightarrow{\quad rcom_k, V \quad} & \\
res_k \leftarrow cr_k + e_k \bmod q & \xleftarrow{\quad c \quad} & c \in_R \mathbb{Z}_2 \\
z_k \leftarrow cx_k + y_k \bmod q & \xrightarrow{\quad res_k, z_k \quad} & g_1^{res_1} g_2^{res_2} \stackrel{?}{\equiv} Vm'^c \bmod p \\
& & g''^{res_k} h''^{z_k} \stackrel{?}{\equiv} rcom_k com_k^c \bmod p \\
\text{END REPEAT} & &
\end{array}
$$

Fig. 2: Step 2 of the Fingerprinting Protocol

Step 3: Now $\mathcal{B}$ additionally computes quadratic residue commitments on the same values r_{kj}. As mentioned, these are needed as input to the embedding procedures. We denote them by $com'_{kj} = BC^{QR}(x'_{kj}, r_{kj})$ for $x'_{kj} \in_R \mathbb{Z}_n^*$, where n is a Blum integer chosen by the buyer. $\mathcal{B}$ sends these commitments to $\mathcal{M}$ and proves in zero-knowledge that the contents in each pair (com_{kj}, com'_{kj}) are equal. Since one can only commit to bits when using BC^{QR}, the equality proof implies that the values r_{kj} in Step 2 were binary. An efficient proof can again be carried out by fairly standard techniques. For instance, one can see two pairs of commitments BC^{DL} and BC^{QR}, where both commitments in one pair contain "0" and in the other pair "1", as a cryptographic capsule and proceed similar to [Ben87]. This is a proof with one-bit challenges and thus the least efficient part of our protocol. However, even if one only compares it with the simplest embedding procedure that might follow, fingerprinting without collusion tolerance as in [PS96], one sees that quadratic residue commitments must be made on a portion of the data item significantly larger than the word emb to be embedded into it (so that the resulting changes are small). Thus the complexity of our last step is not larger than that of embedding.

4.3 Identification

After finding a redistributed copy of the data item, $\mathcal{M}$ tries to identify a traitor as follows:

Step 1: $\mathcal{M}$ extracts a value $emb = (r_1, r_2)$ from the redistributed data item using the extraction algorithm from the underlying embedding scheme. This pair is (is, s) with $s \neq 0$. $\mathcal{M}$ computes $m' \equiv g_1^{is} g_2^s \bmod p$ and retrieves $coin'$, $text$, and sig_{text} from the purchase record of the corresponding data item. If he does not find the coin identifier m', he gives up (the collusion tolerance of the underlying code may be exceeded). Otherwise he sends i to $\mathcal{RC}$.

Step 2: $\mathcal{RC}$ searches in its registration database for a buyer who is registered under the value $h_1 \equiv g_1^i$. It retrieves the values $(ID_{\mathcal{B}}, sig_{coin})$ and sends them to $\mathcal{M}$. Note that $\mathcal{M}$ can enforce $\mathcal{RC}$'s cooperation, see below.

Step 3: $\mathcal{M}$ verifies the signature sig_{coin} on h_1.

Enforced Identification This is a special case in identification if $\mathcal{RC}$ refuses to reveal the information requested by $\mathcal{M}$:

Step 1: $\mathcal{M}$ sends $proof_1 = (coin', (i, s))$ to an arbiter $\mathcal{A}$.

Step 2: $\mathcal{A}$ verifies the validity of $coin'$ using the algorithm from Step 1 of fingerprinting and that $m' \equiv g_1^{is} g_2^s \bmod p$. If this is wrong, $\mathcal{A}$ rejects $\mathcal{M}$'s claim. Otherwise she sends i to $\mathcal{RC}$ and requests the values $(ID_{\mathcal{B}}, sig_{coin})$. Then $\mathcal{A}$ verifies them as $\mathcal{M}$ does in Step 3 of the identification.

4.4 Trial

Now $\mathcal{M}$ tries to convince an arbiter $\mathcal{A}$ that $\mathcal{B}$ redistributed the data item bought under the conditions described in $text$. The values $ID_{\mathcal{B}}$ and $text$ are common inputs.

Step 1: $\mathcal{M}$ sends to $\mathcal{A}$ the proof string

$$proof = ((i, sig_{coin}), (coin', sig_{text}), s).$$

Step 2: $\mathcal{A}$ computes $h_1 \equiv g_1^i \bmod p$ and verifies that sig_{coin} is a valid signature on h_1 with respect to $ID_{\mathcal{B}}$. If yes, it means that i, the internal structure of an account number h_1 for which $\mathcal{B}$ was responsible, has been recovered by $\mathcal{M}$ and thus, as we will see, that $\mathcal{B}$ has redistributed some data item. Note that i alone is not enough evidence for $\mathcal{A}$ to find $\mathcal{B}$ guilty of redistributing a data item under the specific text, $text$.

Step 3: $\mathcal{A}$ verifies the validity of $coin'$, that $m' \equiv g_1^{is} g_2^s \bmod p$ holds, and the signature sig_{text} on the disputed text using the test key pk_{text} contained in $coin'$. These verifications imply that if the accused buyer owned this coin, she must have spent it in the disputed purchase on $text$. Now $\mathcal{A}$ must verify that this coin belongs to $\mathcal{B}$. It is not possible to do so by only showing the link between the coin and the withdrawal (which could be fixed by a signature from $\mathcal{B}$ under $\mathcal{RC}$'s view), because a collusion of $\mathcal{M}$ and $\mathcal{RC}$ could forge such a link. (Interested readers can find the attack in Appendix A.) Thus $\mathcal{A}$ performs the following last step where $\mathcal{B}$ is required to take part.

Step 4: $\mathcal{A}$ asks $\mathcal{B}$ whether she has withdrawn another valid coin, i.e., a tuple $coin^* = (m^*, pk^*_{text}, \sigma^*)$ with $pk^*_{text} \neq pk_{text}$ using the one-time account h_1. If yes, $\mathcal{B}$ has to show the representation of m^*, i.e., a value s^* such that $m^* \equiv g_1^{is^*} g_2^{s^*}$. If $\mathcal{B}$ can do that, then $\mathcal{A}$ decides that $\mathcal{RC}$ is guilty, otherwise $\mathcal{B}$.

5 Security of the Construction

We now present detailed proof sketches of our construction. We assume that all the underlying primitives are secure. The merchant's security only relies on the security of the underlying embedding scheme, the buyer's on standard cryptographic assumptions. The security for the registration center needs the restrictiveness of the blind signature scheme, and we will make a precise assumption for this. Anonymity is information-theoretic.

5.1 Security for the Merchant

Due to the properties of the underlying embedding scheme, we can assume that whenever the maximum tolerated size of a collusion is not exceeded, and the collusion redistributes a data item sufficiently similar to the original, then $\mathcal{M}$ can extract a value emb that belongs to a traitor with very high probability. More precisely emb is the value to which the traitor could open the final quadratic-residue commitments given in the corresponding purchase.

The zero-knowledge proofs in fingerprinting guarantee that this value emb is a pair (r_1, r_2) such that $g_1^{r_1} g_2^{r_2} \equiv m' \neq 1$, where m' is the identifier of the coin used in this purchase. Thus $\mathcal{M}$ can retrieve a valid $coin'$ and the pair $(i, s) \equiv (r_1/r_2, r_2)$. This enables him to ask $\mathcal{RC}$ for identification and, in the worst case, have it enforced by $\mathcal{A}$. Moreover, he can retrieve $text$ and a valid signature sig_{text}. Together with the values that $\mathcal{RC}$ must return, $\mathcal{M}$ therefore obtains a valid proof string $proof$ and passes the first three steps of the trial. This is sufficient because the Step 4 of the trial only concerns $\mathcal{A}$'s decision on whether $\mathcal{RC}$ or $\mathcal{B}$ is guilty.

$\mathcal{M}$ is also protected from making wrong accusations (and thus possibly damaging his reputation): Even if there are more than the tolerated number of traitors, $\mathcal{M}$'s verifications in identification guarantee that whenever he makes an accusation he will not lose in the trial.

5.2 Security for the Buyer

Consider an honest buyer $\mathcal{B}$ and a trial about the purchase on a specific text, $text$, for which $\mathcal{B}$ has not revealed the corresponding data item. She is secure if the attackers cannot convince an honest arbiter $\mathcal{A}$ in this trial, even if the other parties collude and obtain other data items that she bought (active attack). Such situations occur, e.g., if $\mathcal{B}$ is allowed to redistribute another item after a certain period of time.

Step 2 of the trial guarantees that $\mathcal{B}$ is only held responsible for one of her own one-time account numbers $h_1 \equiv g_1^i$, and that the attackers must know i.

First it is shown that the attackers cannot find i unless they obtain the result of a purchase where $\mathcal{B}$ has used a coin $coin^*$ withdrawn from the account h_1. The only knowledge the attackers can otherwise obtain about i in our protocol is: (1) h_1 itself, (2) the proof of knowledge of i in registration, and (3) the commitments on $emb = (is, s)$ and two zero-knowledge proofs in fingerprinting. Additionally, they might obtain information in embedding, but by definition of secure embedding this is not the case. If the proofs are actually zero-knowledge and the commitments semantically secure, computing i from all this information is as hard as computing it from h_1 alone, i.e., as computing a discrete logarithm. (If we use Schnorr identification and a similar proof in fingerprinting, security relies on the joint security of these identification protocols against retrieval of the secret key.)

Hence the only way for the attackers to find i is in fact to obtain the resulting data item in a purchase where $\mathcal{B}$ has used $coin^*$ based on h_1. Let $text^*$ be the text describing that purchase. By the precondition, we know that $text^* \neq text$. The secret key sk^*_{text} corresponding to pk^*_{text} in $coin^*$ is known only to $\mathcal{B}$, and $\mathcal{B}$ reveals no information about it during registration, fingerprinting, and redistribution except making one signature on $text^*$ with it. As we assume that the underlying signature scheme is secure against active attacks, this does not help the attackers to forge a valid signature with respect to pk^*_{text} on $text$.

Now, even if the attackers are a collusion of $\mathcal{M}$ and $\mathcal{RC}$ and succeed in constructing a wrong coin $coin_A$ with a self-made pk^A_{text} which passes all $\mathcal{A}$'s verifications so far, $\mathcal{B}$ wins by showing $coin^*$ with $pk^*_{text} \neq pk^A_{text}$.

5.3 Security for the Registration Center

An honest registration center $\mathcal{RC}$ should never be found guilty by an honest arbiter $\mathcal{A}$. This could happen in two cases:

1. In enforced identification if $\mathcal{M}$ can send a value $proof_1$ that convinces $\mathcal{A}$, but $\mathcal{RC}$ cannot reveal the required registration data under the identity i.
2. In a trial if the attackers (a collusion of $\mathcal{M}$ and $\mathcal{B}$) can generate two valid coins, $coin'$ and $coin^*$, withdrawn from the same account h_1, i.e., their representations correspond to the same h_1.

To exclude both cases, we need the restrictiveness of the underlying blind signature scheme. The corresponding assumption in [Bra94] is fairly informal, so we formalize it in a version that suffices for our purpose. Note that although this version seems closest to Brands' formulation, for his payment system a weaker assumption would be sufficient, and the corresponding definition in [FY93] is also of that weaker form: For the payment system one only needs that all the coins the attackers can construct correspond to some account of the attackers. For Case 2 of $\mathcal{RC}$'s security, we need that the attackers cannot even transform coins between their own accounts. (In [Bra94], Definition 5, this is somehow implicit because a one-to-one correspondence between constructed coins and withdrawal protocols is assumed a priori.) We only need the case with one-time accounts, but we formulate the general case, and also for any number of generators. We first introduce some notation:

- Let gen_{group} be an algorithm that generates a group from the given family (see Section 4) and a fixed number $n + 1$ (in our concrete case $n = 2$) of random generators g and $g_1, ..., g_n$. Its output $(p, q, g, g_1, ..., g_n)$ is denoted by $desc$.
- Let gen_{key} be the key generation algorithm that takes as input a value $desc$ and outputs a key pair (sk, pk), here (x, g^x).
- Let $valid(desc, pk, (m', par, \sigma'))$ be the predicate for validity of a coin as in Step 1 of fingerprinting (where par may be an arbitrary value in the place of pk_{text}).

– By the predicate *repr* we denote that a vector $I = (i_1, ..., i_n) \in \mathbb{Z}_q^n$ is the representation of a message $m \in G_q$, i.e.,

$$repr(m, I) :\Longleftrightarrow\ m = g_1^{i_1} ... g_n^{i_n}.$$

– $blindsig$ is the protocol between a signer $\mathcal{S}$ and a (dishonest) recipient $\mathcal{R}$ shown in Figure 1. Signer and recipient are interactive probabilistic polynomial-time algorithms, and in our case $\mathcal{S}$ always follows the protocol and $\mathcal{R}$ need not. The signer inputs sk and a message m, and the recipient has an arbitrary auxiliary local variable aux as input; it models the attacker's memory between protocol executions. The signer has no final output, while the recipient outputs an update aux'' of its auxiliary variable. If the recipient wants to continue the attack, we also let him immediately output the message m'' that the signer should sign next, and we also allow another output o''. We write one such interaction as

$$((aux'', m'', o''), -) \leftarrow blindsig(\mathcal{R}(aux), \mathcal{S}(sk, m)).$$

We now consider a dishonest recipient $\mathcal{R}$ who executes the withdrawal protocol with an honest Brands signer $\mathcal{S}$ polynomially many times. In addition, $\mathcal{R}$ has to show a representation I_j of each input message m_j. (Note that we make the assumption with actual outputs in places where the real protocols only have proofs of knowledge.) At the end, $\mathcal{R}$ outputs valid signatures, again with representations of the signed messages. The assumption (or, implicitly, the definition of strong restrictiveness) is that all the output representations are scalar multiples of input representations, and in a kind of one-to-one mapping. To express this easily, let $< \ldots >$ be a notation for a multiset (i.e., unordered like a set but possibly with repetition). We use $\subseteq$ to denote the subset relation for multisets, i.e., each element on the left must also occur on the right with at least the same multiplicity. Finally, for any vector (representation) $I \neq 0$ let $\overline{I}$ denote the line it generates, i.e., the set of its scalar multiples.

Assumption 1 (General Strong Restrictiveness) *Let $\mathcal{R}$ denote a probabilistic polynomial-time interactive algorithm, Q a polynomial and l a security parameter. Then for all Q and for all $\mathcal{R}$ that interact with the Brands signer for $k = Q(l)$ times:*

$$
\begin{aligned}
&P(\neg(<\overline{I'_1},\ldots,\overline{I'_{k'}}> \subseteq <\overline{I_1},\ldots,\overline{I_k}>)\\
&\quad \wedge \forall j=1,\ldots,k:\ repr(m_j,I_j)\\
&\quad \wedge \forall j=1,\ldots,k':\ (repr(m'_j,I'_j) \wedge valid(desc,pk,(m'_j,par_j,\sigma'_j)))\\
&\quad \wedge \textit{all pairs } (m'_j,par_j) \textit{ are different}\\
&::\quad desc \leftarrow gen_{group}(l);\ k := Q(l); (sk,pk) \leftarrow gen_{key}(desc);\\
&\quad ((aux_1,m_1,I_1),-) \leftarrow \mathcal{R}(l,desc,pk);\\
&\quad ((aux_2,m_2,I_2),-) \leftarrow blindsig(\mathcal{R}(aux_1),\mathcal{S}(sk,m_1));\\
&\qquad\quad \vdots\\
&\quad ((aux_k,m_k,I_k),-) \leftarrow blindsig(\mathcal{R}(aux_{k-1}),\mathcal{S}(sk,m_{k-1}));\\
&\quad (aux_{k+1},-) \leftarrow blindsig(\mathcal{R}(aux_k),\mathcal{S}(sk,m_k));\\
&\quad (k',((m'_1,par_1,\sigma'_1),\ldots,(m'_{k'},par_{k'},\sigma'_{k'})),\ (I'_1,\ldots,I'_{k'}))\ \leftarrow \mathcal{R}(aux_{k+1})\,)\\
&\le 1/poly(l).
\end{aligned}
$$

Next we return to the security of fingerprinting and consider a successful attacker $\mathcal{R}^*$ against $\mathcal{RC}$'s security. As $\mathcal{RC}$ never uses its secret key except in the blind signature protocol within registration, $\mathcal{R}^*$ can be seen as an attacker against the blind signature protocol with $n = 2$. The only difference to the scenario in Assumption 1 up to the choice of aux_{k+1} is that $\mathcal{R}^*$ need not output the values I_j, but instead gives a proof of knowledge of i_j in the specific representation $m_j = g_1^{i_j} g_2$. We combine $\mathcal{R}^*$ with the extractor for these proofs of knowledge to obtain another attacker $\mathcal{R}$ that also outputs I_j, so that Assumption 1 applies.

We now consider the two cases of how $\mathcal{RC}$ could be found guilty by an honest arbiter with the new attacker $\mathcal{R}$.

In Case 1, the attackers need a valid coin $coin' = (m', pk_{text}, \sigma')$ and a representation $I = (is, s)$ of m' such that no withdrawal with respect to $h_1 = g_1^i$ was performed. (If $\mathcal{RC}$ does find such a withdrawal it is clear that the retrieved values pass $\mathcal{A}$'s tests.) However, by Assumption 1, I is a scalar multiple of one of the original representations, say I_j. As I_j is of the form $(i_j, 1)$, this means that $i = i_j$ and thus a withdrawal with $h_1 = g_1^i$ was performed.

In Case 2, the attackers must show two valid coins $coin' = (m', pk_{text}, \sigma')$ and $coin^* = (m^*, pk^*_{text}, \sigma^*)$ with $pk^*_{text} \neq pk_{text}$ and representations (is, s) of m' and (is^*, s^*) of m^* with the same i. These output representations lie on the same line $\overline{I}$ with $I = (i, 1)$. By Assumption 1, this $\overline{I}$ and thus $h_1 = g_1^i$ also have to occur twice among the input representations. However, $\mathcal{RC}$ ensures that each account is only used once. This finishes the proof that such a successful attacker cannot exist.

5.4 Anonymity

Due to the properties of the underlying cash system the views of $\mathcal{RC}$ and $\mathcal{M}$ concerning the withdrawal of a coin and its verification are unlinkable.

We now consider the additional information in registration and fingerprinting: Obviously, sig_{coin} in registration is no problem because the only additional information it is based on ($\mathcal{B}$'s secret key) is not used in fingerprinting. Similarly, sig_{text} in fingerprinting is no problem: The only additional information it is based on is sk_{text}. However, an information-theoretic attacker can compute sk_{text} from pk_{text}, which is already transmitted in the coin, and we know that the coin-related data in the underlying cash system are not linkable.

The value emb in fingerprinting, which is based on the same i that is also used in fingerprinting, is hidden in bit commitments. The first commitment scheme used, based on discrete logarithms, is information-theoretically hiding and the second one, based on quadratic residues, is hiding under the Quadratic Residuosity Assumption. Moreover the zero-knowledge protocols used in fingerprinting do not leak information about emb and the embedding operation is assumed not to leak such information either.

Furthermore using one-time accounts implies that $\mathcal{B}$'s different purchases are unlinkable, even if a purchased data item has been redistributed and the information contained in it recovered.

6 Conclusion

We have presented an anonymous fingerprinting scheme where all protocols are explicit and fairly efficient. The complexity is lower than that of any known embedding procedure that might follow, so that the anonymity is currently not the bottleneck for implementation. A disadvantage of this scheme in comparison with the previous one based on general zero-knowledge techniques is that the buyer is needed to carry out a fair trial; we hope to find a scheme that combines explicitness and 2-party trials in the future.

Of course, embedding for any asymmetric scheme is still rather an expensive procedure because commitments on a significant fraction of the data are needed, and the data must be long because otherwise one could not hide enough bits in them, at least if one desires collusion tolerance.

Acknowledgments: We thank Matthias Schunter and Michael Waidner for generous and fruitful discussions.

References

[BCC88] Gilles Brassard, David Chaum, Claude Crépeau: Minimum Disclosure Proofs of Knowledge; Journal of Computer and System Sciences 37 (1988) 156-189.

[BCP88] Jurjen Bos, David Chaum, George Purdy: A Voting Scheme; unpublished manuscript, presented at the rump session of Crypto'88.

[Ben87] Josh Cohen Benaloh: Cryptographic Capsules: A Disjunctive Primitive for Interactive Protocols; Crypto'86, LNCS 263, Springer-Verlag, Berlin 1987, 213-222.

[BKK90] Joan F. Boyar, Stuart A. Kurtz, Mark W. Krentel: A Discrete Logarithm Implementation of Perfect Zero-Knowledge Blobs; Journal of Cryptology 2/2 (1990) 63-76.

[BM97] Ingrid Biehl, Bernd Meyer: Protocols for Collusion-Secure Asymmetric Fingerprinting; STACS 97, LNCS 1200, Springer-Verlag, Berlin 1997, 399-412.

[BMP86] G. R. Blakley, Catherine Meadows, George B. Purdy: Fingerprinting Long Forgiving Messages; Crypto'85, LNCS 218, Springer-Verlag, Berlin 1986, 180-189.

[Bra94] Stefan Brands: Untraceable Off-line Cash in Wallet with Observers; Crypto'93, LNCS 773, Springer-Verlag, Berlin 1994, 302-318.

[BS95] Dan Boneh, James Shaw: Collusion-Secure Fingerprinting for Digital Data; Crypto'95, LNCS 963, Springer-Verlag, Berlin 1995, 452-465.

[CEG88] David Chaum, Jan-Hendrik Evertse, Jeroen van de Graaf: An Improved Protocol for Demonstrating Possession of Discrete Logarithms and some Generalizations; Eurocrypt'87, LNCS 304, Springer-Verlag, Berlin 1988, 127-141.

[CFN94] Benny Chor, Amos Fiat, Moni Naor: Tracing Traitors; Crypto'94, LNCS 839, Springer-Verlag, Berlin 1994, 257-270.

[CHP92] David Chaum, Eugène van Heijst, Birgit Pfitzmann: Cryptographically Strong Undeniable Signatures, Unconditionally Secure for the Signer; Crypto'91, LNCS 576, Springer-Verlag, Berlin 1992, 470-484.

[CP93] David Chaum, Torben Pryds Pedersen: Wallet Databases with Observers; Crypto'92, LNCS 740, Springer-Verlag, Berlin 1993, 89-105.

[Dom98] Josep Domingo-Ferrer: Anonymous Fingerprinting of Electronic Information with Automatic Identification of Redistributors; Electronics Letters 34/13 (1998) 1303-1304.

[FN94] Amos Fiat, Moni Naor: Broadcast Encryption; Crypto'93, LNCS 773, Springer-Verlag, Berlin 1994, 480-491.

[FTY96] Yair Frankel, Yiannis Tsiounis, Moti Yung: "Indirect Discourse Proofs": Achieving Efficient Fair Off-Line E-cash; Asiacrypt'96, LNCS 1163, Springer-Verlag, Berlin 1997, 287-300.

[FY93] Matthew Franklin, Moti Yung: Secure and Efficient Off-Line Digital Money; 20th International Colloquium on Automata, Languages and Programming (ICALP), LNCS 700, Springer-Verlag, Berlin 1993, 265-276.

[GM84] Shafi Goldwasser, Silvio Micali: Probabilistic Encryption; Journal of Computer and System Sciences 28 (1984) 270-299.

[KD98] Kaoru Kurosawa, Yvo Desmedt: Optimum Traitor Tracing and Asymmetric Schemes; Eurocrypt'98, LNCS 1403, Springer-Verlag, Berlin 1998, 145-157.

[NP98] Moni Naor, Benny Pinkas: Threshold Traitor Tracing; Crypto'98, LNCS 1462, Springer-Verlag, Berlin 1998, 502-517.

[Ped92] Torben Pryds Pedersen: Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing; Crypto'91, LNCS 576, Springer-Verlag, Berlin 1992, 129-140.

[Pfi96] Birgit Pfitzmann: Trials of Traced Traitors; Information Hiding, LNCS 1174, Springer-Verlag, Berlin 1996, 49-64.

[PS96] Birgit Pfitzmann, Matthias Schunter: Asymmetric Fingerprinting; Eurocrypt'96, LNCS 1070, Springer-Verlag, Berlin 1996, 84-95.

[PW97a] Birgit Pfitzmann, Michael Waidner: Asymmetric Fingerprinting for Larger Collusions; 4th ACM Conference on Computer and Communications Security, acm press, New York 1997, 151-160.

[PW97b] Birgit Pfitzmann, Michael Waidner: Anonymous Fingerprinting; Eurocrypt'97, LNCS 1233, Springer-Verlag, Berlin 1997, 88-102.

[Sch91] Claus-Peter Schnorr: Efficient Signature Generation by Smart Cards; Journal of Cryptology 4/3 (1991) 161-174.

A Linking a Self-Made Coin to a Correct Withdrawal

In this appendix, we show the attack that motivates the last verification step in the trial, as mentioned in Section 4.4.

Let h_1 be an account number of $\mathcal{B}$ and $coin^* = (m', pk^*_{text}, \sigma')$ (with $\sigma' = (z', a', b', r')$) be the coin that $\mathcal{B}$ withdrew from this account and used in a purchase with the text $text^*$. Note that $a' \equiv a^u g^v = g^{uw+v} \bmod p$ and $b' \equiv b^{su} m'^v = m^{wsu} m'^v = m'^{wu+v} \bmod p$.

Assume now that the attackers are a collusion of $\mathcal{M}$ and $\mathcal{RC}$ and have access to the data item that $\mathcal{B}$ has bought in this purchase. Then they know $emb = (is, s)$, w, x, $view_{withdraw} = (a, b, c)$, r, and the additional signature sig^*_{coin} that $\mathcal{B}$ would give to fix $view_{withdraw}$ before receiving r.

As sig^*_{coin} fixes the values (a, b, c) that can be used together with the given h_1, the attackers want to link a self-made coin $coin'_A$, in particular with a self-made pk^A_{text} for which the attackers know sk^A_{text}, to these values. For this, they randomly select $w_A \in_R \mathbb{Z}_q$ and compute $a'_A \equiv g^{w_A} \bmod p$ and $b'_A \equiv m'^{w_A} \bmod p$. Then they compute $c'_A = hash(m', z', a'_A, b'_A, pk^A_{text})$. Since c is fixed they first compute $u_A \equiv c'_A / c \bmod q$ and then $v_A \equiv w_A - wu_A$ and $r'_A \equiv ru_A + v_A \bmod q$. In this manner they obtain values which pass all verifications by $\mathcal{A}$.

Finally, the attackers can use sk^A_{text} to sign any $text$ they like with respect to this coin.