RZ 2881 (#90829) 11/18/96
Computer Science/Mathematics 14 pages

Research Report

Anonymous Fingerprinting

Birgit Pfitzmann

Institut für Informatik
Universität Hildesheim
Samelsonplatz 1
D-31141 Hildesheim, Germany
email: pfitzb@informatik.uni-hildesheim.de

Michael Waidner

IBM Research Division
Zürich Research Laboratory
Säumerstrasse 4
CH-8803 Rüschlikon, Switzerland
email: wmi@zurich.ibm.com

Anonymous Fingerprinting

Birgit Pfitzmann, Michael Waidner

Abstract. Fingerprinting schemes deter people from illegally redistributing digital data by enabling the original merchant of the data to identify the original buyer of a redistributed copy. Recently, asymmetric fingerprinting schemes were introduced. Here, only the buyer knows the fingerprinted copy after a sale, and if the merchant finds this copy somewhere, he obtains a proof that it was the copy of this particular buyer.

A problem with all previous fingerprinting schemes arises in the context of electronic marketplaces where untraceable electronic cash offers buyers privacy similar to that when buying books or music in normal shops with normal cash. Now buyers would have to identify themselves solely for the purpose of fingerprinting. To remedy this, we introduce and construct anonymous asymmetric fingerprinting schemes, where buyers can buy information anonymously, but can nevertheless be identified if they redistribute this information illegally.

A subresult of independent interest is an asymmetric fingerprinting protocol with reasonable collusion-tolerance and 2-party trials, which have several practical advantages over the previous 3-party trials. Our results can also be applied to so-called traitor tracing, the equivalent of fingerprinting for broadcast encryption.

1 Introduction

Fingerprinting schemes are cryptologic mechanisms for the copyright protection of digital data. They do not rely on tamper-resistance, i.e., it is assumed that the buyers obtain the data digitally and can in principle copy them. Buyers who abuse this possibility by illegitimately redistributing the data are called traitors. Fingerprinting schemes discourage traitors by enabling the original merchant of the data to identify the traitor who originally bought the copy.

1.1 Known Classes of Fingerprinting Schemes

Conventional fingerprinting schemes, called symmetric here, essentially work as follows: The merchant prepares a slightly different “copy” of the data item for each buyer. If he finds a redistributed data item, he finds out to which of the copies sold it corresponds. This concept was introduced in [Wagn 83]. For examples of how one can make imperceptible differences in copies, and more references, see [ZhKo 95, BoRD 95, CKLS 96]. Fingerprinting became a cryptologic topic with the problem of collusion-tolerance: What if several traitors collude and compare their copies to find and then eliminate differences? This problem was first considered in [BlMP 86]; solutions that can tolerate larger collusions were presented in [BoSh 95].

In these symmetric schemes, the merchant finds out the identity of a traitor, but cannot convince any third party of this treachery, because he does not find anything in the redistributed copy that he could not have made up himself. In contrast, in asymmetric schemes, introduced in [PfSc 96], the merchant obtains a proof of the treachery. For this, fingerprinting must be an interactive protocol between the buyer and the merchant where the buyer also inputs a secret, and the merchant does not see the fingerprinted copy that this buyer obtains. Only if he finds this copy after a redistribution, he can extract the proof. The same collusion-tolerance as in the symmetric schemes in [BoSh 95] was achieved for asymmetric fingerprinting in [PfWa 96, BiMe 96].

So-called traitor tracing is the equivalent of fingerprinting for cryptologic keys. It was introduced in [ChFN 94] for broadcast encryption, i.e., for situations where the real data, e.g., a Pay-TV movie, are broadcast in encrypted form, and only the keys needed to decrypt the data are sold. Now a different personal key is sold to each buyer; the encryption scheme is adapted so that all the personal keys can be used to decrypt the same ciphertext. The schemes in [ChFN 94] already achieve good collusion-tolerance. (Actually, these techniques were the basis for collusion-tolerant normal fingerprinting.) Asymmetric traitor tracing, introduced in [Pfit 96], analogous to asymmetric fingerprinting, guarantees that the merchant obtains a proof of treachery if he finds a redistributed key. Reasonable collusion-tolerance for asymmetric traitor tracing was achieved in [PfWa 96], too.

One type of scheme that so far only exists for traitor tracing [PfWa 96], but not for normal fingerprinting, is an asymmetric scheme with reasonable collusion-tolerance and 2-party trials. A 2-party trial means that the merchant can just take his proof and convince any arbiter with it, whereas in a 3-party trial, the buyer also has to take part. One advantage of 2-party trials is that one need not find the buyer to carry out the trial. However, this advantage is minor, because in a real trial, the buyer would have to be notified anyway and non-technical points would have to be discussed, e.g., whether someone could have stolen the data item from an honest buyer. More importantly, in a 3-party trial, the buyer also still has to find some secrets, which means that he should not have forgotten the password needed to use them, or died without leaving it to someone else. Additionally, one has to take care with multiple trials about the same data item, because the buyer might have to divulge something about his secrets in each trial. Finally, 2-party trials are much easier to use as subprotocols in other schemes, as we will see below.

1.2 Anonymous Fingerprinting

Electronic marketplaces are supposed to offer similar privacy as current marketplaces. Thus it should be possible to buy cheap objects like books, pictures, and pieces of music anonymously. This becomes even more important if one buys individual articles of what would have been a book or a magazine on paper, because the choice of articles gives a lot of information about a person's lifestyle, habits, etc. For such purposes, anonymous networks, anonymous cash-like payment systems, and even protocols for anonymous, but secure exchange of payment and goods exist, see, e.g., [Chau 81, Chau 85, BüPf 90] for early examples and [Bran 94] for an efficient anonymous off-line payment system with identification of doublespenders.

It would be a pity if all this anonymity were destroyed just because the buyers had to identify themselves for the purpose of fingerprinting or traitor tracing. However, this undesirable situation would occur with all previous symmetric and asymmetric fingerprinting schemes: The buyer has to identify himself for (key) fingerprinting during a purchase, and thus for each particular data item bought, e.g., one picture in fingerprinting or one Pay-TV movie in traitor tracing.

The goal in this paper is therefore to carry out fingerprinting anonymously, but nevertheless to enable the merchant to identify traitors later. This possibility of identification will *only* exist for traitors, whereas honest buyers will remain anonymous. All our schemes will be asymmetric, i.e., the merchant can also convince any third party that a particular person was a traitor.

1.3 Our Results

In Section 2, we introduce the exact model of anonymous fingerprinting and discuss some variants. In Section 3, we present a construction framework for anonymous fingerprinting that makes certain assumptions about an underlying fingerprinting scheme. In Section 4, we show how this framework can be instantiated with some existing fingerprinting and traitor tracing schemes, and why a gap remains. In Section 5, we fill this gap by constructing a scheme for collusion-tolerant asymmetric fingerprinting with 2-party trials, using Reed-Solomon-codes for low-rate error-and-erasure decoding. This scheme is of interest in its own right, too. Some conclusions are drawn in Section 6.

2 Precise Model

We assume that at the start of our scheme, each buyer already has a key pair (sk_B, pk_B) of a digital signature scheme, so that the public key can serve as a digital identity. Thus we can require a buyer to sign something under his identity in a protocol.

For modularity, we also require buyers to register specifically for the fingerprinting scheme under their digital identity. This allows us to make the protocols of the fingerprinting scheme concrete, without fixing how the validity of the initial digital identity is verified. In some situations, this registration could be joined with the initial establishment of the digital identity. The parties where registration can be done are called registration centers. A reasonable choice is the buyer's bank, in particular if the fingerprinted data are paid with anonymous digital cash, because the buyer has to register with a bank anyway and will only be anonymous among this bank's clients. We do *not* require the registration centers to be particularly trusted by any other party; in the strongest of our models, the only bad thing a registration center can successfully do is refuse registration.

Thus we have four types of parties: Merchants, buyers, registration centers, and arbiters who should be convinced in trials. Technically, the role of arbiter should not be restricted, i.e., it should be possible to convince anyone as long as they know a few specific public keys. We can still get quite a number of different definitions, depending on how active the registration centers and arbiters have to be, and whether the merchants and buyers have to trust the registration centers for any or many requirements. We are primarily interested in cryptologic solutions with minimal trust (where a cheating registration center can only refuse registration), but we also mention weaker models.

We only present a detailed definition for fingerprinting schemes, not for traitor tracing schemes. We follow the style of [PfSc 96], but introduce somewhat less explicit notation for brevity.

Definition 1 (Components of anonymous fingerprinting). An anonymous fingerprinting scheme consists of seven protocols. Each interactive algorithm for a party in a protocol is polynomial-time and may be probabilistic, and it may produce an output *failed* to indicate that the protocol could not be finished in the normal way. Security parameters k for computational security, σ for error probabilities in information-theoretic properties, and *coll_size* for the maximum number of colluding traitors are common inputs.

- *Registration center key distribution*: A registration center generates a key pair (sk_{RC}, pk_{RC}), typically of an underlying signature scheme, and distributes pk_{RC} reliably to all merchants, arbiters, and the buyers that might register at this center.
- *Registration* is a two-party protocol between a buyer and a registration center. The common inputs are the buyer's identity (whose validity the registration center must verify outside the protocol), the registration center's public key pk_{RC}, and possibly an upper bound N_B on the number of purchases that the buyer can make based on one such registration. The registration center's secret input is its secret key. We call the outputs the registration center's and the buyer's registration record.
- *Data initialization* is an algorithm the merchant carries out for each data item to be sold. He inputs the data item and possibly an upper bound N_M on the number of times he will sell this data item. (This protocol could be included into the first sale, i.e., the first execution of fingerprinting, but it

is often useful to consider common precomputations separately.) The output is called the merchant's initial data record.

- *Fingerprinting* is a two-party protocol between a merchant and an anonymous buyer. The merchant secretly inputs the data item and the corresponding initial data record, and, not necessarily secretly, the public key of the registration center with whom the buyer registered. The buyer inputs his registration record or an update of it, and both input a common text that describes what this purchase is about.

  The output for the merchant is called a purchase record. The main output for the buyer is the fingerprinted data item; he may also obtain an update on his registration record (e.g., a purchase counter is increased and in schemes with 3-party trials, evidence is stored).

- *Identification* is either an algorithm for the merchant alone or a two-party protocol between the merchant and the registration center. The merchant's input is a redistributed data item whose original buyer he wants to identify, the original version of this data item, the initial data record, and all the purchase records for this data item. If the registration center takes part, its input is its registration records.

  The output for the merchant should be the identity of a buyer, the text used in the particular purchase, and another string called *proof*.

- *Enforced identification*. For cases where the registration center is needed in identification, but refuses to cooperate, there must be a 3-party version of identification that includes an arbiter. The merchant should get the same outputs as in identification, and the arbiter either obtains the output *ok* or *center_guilty*, which denotes that the arbiter has noticed misbehavior by the registration center.

- *Trial* is a two- to four-party protocol between at least the merchant and an arbiter, and possibly a buyer and a registration center. The common inputs are the identity of the accused buyer and the text denoting the disputed purchase. The merchant also inputs the string *proof* obtained in identification. If the registration center takes part, it inputs the registration record of this buyer, and if the buyer takes part, he inputs his current updated registration record (typically just the evidence from the disputed purchase).

  The main output is the arbiter's result. It may be *guilty*, which means that the arbiter finds the buyer a traitor, or *not_guilty*, which means that he rejects the accusation. In some systems, the output can also be *center_guilty*, which means that no decision between the merchant and the buyer could be reached because of wrong behaviour of the registration center. ♦

In the following, we describe the security requirements on such a scheme. All should be fulfilled under active attacks, too. Generally, an active attack means that the attackers can influence the sequence of protocols the honest users carry out and the user inputs (e.g., the texts), obtain some outputs from the users (e.g., whether a protocol failed or not), and behave maliciously during the protocol executions.

Definition 2 (Effectiveness).

- Correct case. Registration and data initialization should end successfully, i.e., not with the output *failed*, if the parties in the given protocol execution are honest. Similarly, fingerprinting should end successfully if the merchant, the buyer, and the buyer's registration center are honest, and the fingerprinted data item should be sufficiently similar to the data item input by the merchant. Similarity can be formalized by a given relation as in [PfSc 96].

- No jamming by registration center. Even for a cheating registration center, it is infeasible to carry out registration with a buyer such that it ends successfully, but nevertheless an execution of fingerprinting between this buyer and an honest merchant will fail later. ♦

The second property is one of those that define minimal trust in the registration center. Of course, it cannot be avoided that a cheating registration center refuses or messes up registration altogether. However, if the buyer notices this by the output *failed*, it is no problem: He can register at another center. It would only be a problem if fingerprinting failed later and the buyer and the merchant would not know whether to blame each other or the center. The name "jamming" was taken from the consideration of similar frauds by arbiters in arbitrated authentication schemes in [DeYu 91].

Definition 3 (Integrity).

- Security for the merchant. For any algorithm $\tilde{B}$ of the cheating buyers that buys at most *coll_size* copies of a certain data item (i.e., engages in at most *coll_size* executions of fingerprinting for it) and then produces another copy sufficiently similar to the original for the merchant to feel cheated, the merchant will successfully identify a buyer, i.e., obtain a valid digital identity as an output in identification, together with a text used and a string *proof*, and then win a trial with any honest arbiter. Similarity is defined by a second relation as in [PfSc 96], and $\tilde{B}$ may carry out any other transactions, such as additional registrations and buying other data items, in between as part of its active attack.

  This should hold even if the registration centers are cheating, i.e., $\tilde{B}$ also comprises them. In this case, the protocol for enforced identification may be needed if normal identification failed, and the output for the arbiter in either this protocol or the trial may be *center_guilty*, instead of *guilty* in the trial.[1]

- Protecting the merchant from making wrong accusations. As the merchant will usually damage his reputation if he accuses a buyer and then loses the trial, we require that this does not happen to honest merchants. Thus, even if there are more than *coll_size* traitors, it should be infeasible for the other participants to make up a data item such that identification succeeds, and then a trial with an honest arbiter leads to the output *not_guilty*.

- Security for the buyer. Honest buyers are not found guilty in trials. More precisely, if a buyer only takes part in the prescribed protocols and keeps their results secret (in particular, the data item bought), then, no matter what the other parties do, an honest arbiter will not obtain the output *guilty* in a trial where he entered the identity of this buyer. Even if the other parties can adaptively obtain some data items this buyer bought, selected by the texts used in the corresponding execution of fingerprinting, the buyer will not be found guilty for any other texts.

- Security for registration centers. In schemes with strong security for the merchant, i.e., where an arbiter may decide *center_guilty*, honest registration centers require that honest arbiters never decide this about them. ♦

In a weaker version of security for the merchant, the requirement would only hold if at least the registration centers the dishonest buyers registered with are honest. A similar weak version of the security for the buyer is not desirable, because being wrongly found guilty as a traitor is a fate much worse than losing some revenue.

---

1 A stronger requirement that it is always a buyer who is identified would not make much sense: If a registration center colludes with some traitors, it can be regarded as one of them; actually, identifying a cheating registration center is more important than identifying a normal buyer and the merchant is more likely to get compensation.

Finally, we come to the privacy requirements. We only make them explicitly for buyers, corresponding to the usual model of payer anonymity in digital payment systems. However, the identity of the merchant is not needed anyway, neither above nor in other types of fingerprinting.

Definition 4 (Anonymity). Nothing about the purchase behaviour of honest buyers becomes known to any other party, except, if the registration center cooperates, for facts that can simply be derived from the knowledge of who registered and for which number of purchases, N_B, and at what time protocols are executed. This should even hold for the remaining purchases if the other parties can adaptively obtain some data items this buyer bought. ♦

The exception cannot be avoided. For instance, if the first person who registers buys something before anybody else registers, the merchant and the registration center together naturally know who it was. Furthermore, the definition assumes, like that of anonymous payment systems, that the underlying communication does not identify the buyers. The definition is otherwise very strict. For instance, it implies that the merchant cannot learn whether a particular person bought a particular data item by accusing him unjustly of redistribution.

We could also define weaker versions of anonymity, in particular k-out-of-n traceability and linkability. Similar models have been considered with payment systems, often without distinguishing them. Some types of fingerprinting with weak anonymity can be implemented quite easily and without any real additional cryptology, but we omit these constructions in favor of stronger ones.

3 Construction Framework for Full Anonymity

During fingerprinting, the buyer has to input identifying information that will be embedded into the data item; we call it *emb*. The merchant must be convinced that this information is correct, but without learning more about it. Hence a construction has to address two major issues:

- Relating the identifying information *emb* to the public key of the registration center, so that the merchant has a starting point for the verification that does not identify the buyer and does not make purchases linkable, together with a minimum-knowledge verification procedure.
- A mechanism for the merchant to extract *emb* from a redistributed data item. This is not trivial, because in most non-anonymous schemes, information is not simply "extracted" from the data item found, but derived in combination with other information or in interaction with an accused buyer, each of which is more complicated here.

In this section, we show a construction framework that includes a solution to the first issue, but assumes a subprotocol that solves the second issue.

Construction 1 (Framework for anonymous fingerprinting.) We only show those protocols where anything interesting happens at this level of abstraction.

- In registration, the buyer selects a pseudonym, i.e., a key pair (sk_B^*, pk_B^*) of a signature scheme, and signs under his normal identity that he will be responsible for this pseudonym. He obtains a certificate $cert_B$ from the registration center, i.e., a signature with sk_{RC} on pk_B^*. Intuitively, this certificate means that the registration center declares that it knows the identity of the buyer who chose this pseudonym.
- In fingerprinting, the anonymous buyer secretly computes a signature on the text identifying the purchase, $sig := sign(sk_B^*, text)$. The entire value to be embedded is $emb := (text, sig, pk_B^*, cert_B)$.

This buyer hides this value in a commitment (see [BrCC_88]), sends the commitment to the merchant, and proves the validity of the hidden signature and certificate in zero-knowledge.

Instead of embedding *emb* directly, the buyer can encrypt it, send the ciphertext to the merchant, and commit to and embed the key, which may be much shorter. The zero-knowledge proof now refers to the value obtained by decrypting the given ciphertext with the hidden key.

- In identification, the merchant extracts *emb*. He sends $proof_1 := (text, sig, pk_B^*)$, which proves that the owner of this pseudonym has redistributed the data item corresponding to *text*, to the registration center and asks for identification. If the registration center refuses, the merchant shows $proof_1$ to an arbiter, together with $cert_B$ to prove that the registration center knows the corresponding identity. Thus, in enforced identification, the registration center either has to identify or will be found guilty. Moreover, the registration center has to send the buyer's signature that he is responsible for this pseudonym. This signature and $proof_1$ constitute *proof*. The merchant verifies all the values before making an accusation.

  In the version with encryption, the merchant tries to decrypt the ciphertexts from all the purchase records for this data item. He verifies the resulting cleartexts as above, and uses the first that fulfils the criterion.

- In a trial, the arbiter first verifies the accused buyer's signature that he is responsible for the pseudonym pk_B^*, and then that *sig* is a valid signature on *text* corresponding to this pseudonym.

Theorem 1. If all the underlying primitives are secure, the construction framework yields a provably secure anonymous fingerprinting protocol. ♦

The proof is quite straightforward and omitted here. In particular, it is assumed for the security and anonymity of the buyer that the scheme used for embedding does not leak information about *emb*, and for the security of the merchant that extracting will in fact recover the embedded value if there are at most *coll_size* traitors.

4 Instantiation with Known Fingerprinting Schemes

We now identify existing fingerprinting schemes that offer the combination of embedding and extracting needed in Construction 1. We also describe some details of other fingerprinting schemes, because they are helpful for understanding the new construction in Section 5.

For the cryptologic aspects of fingerprinting, it is typically assumed (starting with [BlMP 86, BoSh 95]) that a marking scheme is given, i.e., a data-type-dependent scheme for hiding individual bits in data items. Each mark is a part of the data item for which 2 versions exist. In data initialization, the merchant probabilistically selects a tuple of marks for the given data item. Each fingerprinted data item can now be described by a binary codeword: the i-th bit denotes which version of the data is used in the i-th mark. It is assumed that traitors can only notice and delete marks by comparing their copies. More precisely, the Marking Assumption [BoSh 95] states that if the codewords of all traitors agree in the i-th bit, any redistributed copy they make will correspond to a word with the same i-th bit.

A consequence of the Marking Assumption is that in any redistributed data item produced by at most *coll_size* traitors, the merchant will find a word that has at least l / *coll_size* symbols in common with the codeword of at least one traitor. (If the traitors delete a mark they have identified, instead of using one of the 2 versions, the merchant arbitrarily sets the corresponding bit in the word to 0 or 1.) The merchant now has to derive some real information; this can be seen as a problem of error correction for far more errors than correct symbols. We now consider how different fingerprinting schemes deal with this problem, and whether they offer the direct extraction we need (+/–):

+ Symmetric schemes with (almost) no collusion-tolerance: If there is no collusion at all, the marking assumption implies that the whole codeword of the traitor remains intact. Hence it can simply be extracted. Some schemes do not assume traitors to be clever and hope that the majority of one word will still be intact, so that a normal error-correcting code can be used.

– Symmetric collusion-tolerant schemes [BlMP 86, BoSh 95]: Essentially, the merchant looks through the list of the codewords he has used and checks which of them has *l/coll_size* symbols in common with the redistributed word. (In fact, a somewhat more complicated code and comparison is used to make it provably unlikely that an honest participant's codeword also has so many symbols in common with the redistributed word.) These schemes cannot be used for embedding and extracting a significant amount of information, because the merchant does not know the codewords that were used, and a list of all possible ones would be exponentially long.

– Asymmetric schemes with 3-party trials also had to address the problem that the codewords used cannot be known to the merchant entirely, because parts of them are needed to make up *proof*, the proof of redistribution, when they are found. The idea in [PfWa 96, BiMe 96] is to make one half of the codeword known to the merchant in fingerprinting and to keep the other half secret. In identification, the merchant first searches a list of the known halves to identify a buyer, whom he accuses. He only has the other half, which should contain *proof*, with a large number of errors, too many to decode efficiently. Thus the accused buyer is now asked to show the real *proof*, and the arbiter compares if it has enough symbols in common with what the merchant found.

  However, this three-party idea cannot be used in the anonymous case, because the merchant does not know whom to accuse before he has found the correct secret, and one cannot ask many buyers to divulge theirs. More technically, we see that *proof* is not actually extracted.

+ Asymmetric collusion-tolerant traitor tracing with 2-party trials [PfWa 96, Section 4] (based on ideas from [ChFN 94]): A code is used where some parts of the codeword must be taken from one traitor as a whole. The entire secret that will be the main part of *proof* is used as many such parts, so that it will come through at least once.

  This scheme can be used for embedding and extracting arbitrary values *emb*: These values are treated just as the main part of the proof was treated above. In the notation of [PfWa 96] for readers familiar with it: *emb* is used as the second-level codewords instead of rid_B. All parts of the scheme that do not deal with embedding and extracting, i.e., the one-way image of rid_B and its signing and verification, are omitted.

For fingerprinting, there seems to be no idea yet how to glue large parts together so that they have to be taken from one traitor as a whole, as in traitor tracing. However, in the following, we will use much smaller parts that will be correct as a whole, and apply error-and-erasure-correcting codes.

5 Collusion-Tolerant Asymmetric Fingerprinting with 2-Party Trials

5.1 Ideas

Recall the basic idea from [BoSh 95] to achieve a certain level of collusion-tolerance among a large number of participants: A concatenated code (called nested in [Blah 83]) is used where the outer words are of length l over the alphabet $\{1, \ldots, q\}$, and the inner code, which is used to encode each symbol of an outer codeword, is a fixed binary code Γ_0 of length $d(q-1)$, where l, d, and q are three parameters that we adapt to our purposes below.

The important property of Γ_0 is that it has a decoding procedure that guarantees that, except with exponentially small probability, an outer symbol that appeared in the codeword of at least one traitor will be extracted in each position. The precise error probability is $2^{-\sigma}$ for all l outer symbols together, if d is chosen as $2q^2(\log_2(2ql) + \sigma)$.

Thus the symbols of the outer codeword are blocks that have to be taken from one traitor as a whole, as desired in the construction idea in Section 4. However, they can only encode a very small number of bits, because the inner code is essentially unary. Thus we proceed in a more complicated way to put several such small pieces together again, i.e., to try and find a certain number that come from the same traitor. For this, we will link known and secret halfsymbols (in contrast to the unconnected known and secret symbols of the words in [PfWa 96, BiMe 96]), so that symbols that disagree on the known halfsymbols can be excluded right away. This leaves us with many erasures, but hopefully few errors, and thus we can hope for efficient decoding. We will do this with Reed-Solomon codes, but we first present the rest of the construction without fixing the code.

5.2 Construction with Generic Code

The following construction is only a scheme for embedding and extracting data. It can either be used in Construction 1 to obtain an anonymous collusion-tolerant fingerprinting scheme, or as a normal collusion-tolerant asymmetric fingerprinting scheme with 2-party trials. For the latter, the values *emb* are selected and treated like the values id_{sym} in Construction 1 of [PfSc 96]: In fingerprinting, *emb* is randomly chosen by the buyer, and a one-way image *im* of it is signed and given to the merchant. Later, having found the preimage *emb* of *im* proves that the merchant found the redistributed data.

Note that in both these applications, Construction 2 and the surrounding scheme are coupled over a *secret* value, *emb*, that must be the same in both schemes, i.e., the same commitment must be used.

We denote the binary length of the values to be embedded as a function $len(k)$ of the computational security parameter, because they are usually cryptologic secrets. The following construction is in terms of four parameters l, d, q_1, and q_2, which will be chosen as polynomial functions of the given parameters k, σ, $coll\_size$, and N_M. Here, l and d will be used for a concatenated code exactly as explained above, and the parameter q for that code will be q_1q_2. We assume that q_1 and q_2 are small powers of 2, say $q_i = 2^{\kappa_i}$. Thus each symbol of the outer code can be represented as the concatenation of two short strings of length κ_1 and κ_2.

We also need an error-and-erasure-correcting code *EECC* of the same length l over an alphabet of size q_2 and of sufficient dimension *dim* to encode the values to be embedded, i.e., $\kappa_2 dim \geq len(k)$. The precise error-and-erasure-correcting properties needed are discussed below.

Construction 2 (Embedding and extracting).

- *Data initialization*. The merchant chooses the marks for the data item. Furthermore, for each of the l positions of the outer code, he chooses a substitution $subst_i$ randomly, i.e., a permutation of the alphabet $\{1, \ldots, q\}$. Recall that the alphabet is small enough for a random permutation to be represented as a table.
- *Embedding*: The merchant's secret inputs are the data item and the initial data record. The commitments that fix the value emb_B to be embedded for the current buyer are a common input.[2] The buyer's secret input is emb_B and the values needed to open the commitments.
  - The merchant secretly selects κ_1 randomly chosen bits for each of the l symbols of the outer codeword. We call them halfsymbols and denote the choice as
  $$halfword\_search_B := (halfsym\_search_{B,1}, \ldots, halfsym\_search_{B,l}).$$
  - Now emb_B is encoded with the error-and-erasure-correcting code *EECC* into l halfsymbols of κ_2 bits each. We call them $halfsym\_emb_{B,1}, \ldots, halfsym\_emb_{B,l}$. The buyer can do this alone if he hides the result in commitments again and proves in zero-knowledge that the computation was correct.
  - The halfsymbols from the merchant and the buyer are mixed into symbols by the operation
  $$sym_{B,i} := subst_i(halfsym\_search_{B,i} \,||\, halfsym\_emb_{B,i}),$$
  where $subst_i$ is the substitution chosen in data initialization for this symbol position. We will see below why this encryption is necessary for the security of the merchant. This step and the following one require secure 2-party computation, because secrets from both parties are used. The outer codeword of this buyer is
  $$word_B := (sym_{B,1}, \ldots, sym_{B,l}).$$
  - Each outer symbol $sym_{B,i}$ is encoded using the inner code Γ_0, and the resulting word is used to fingerprint the data item.
- *Extracting*.
  - For each of the l positions of the outer code, the merchant uses the identification procedure of the underlying code Γ_0 to identify a symbol $sym_{red,i}$ ("*red*" for "redistributed"). He decrypts it using $subst_i^{-1}$ and separates it into its halves of length κ_1 and κ_2, respectively. The resulting outer word is called $word_{red}$, and the word consisting of all the first halves $halfword\_search_{red}$.
  - The merchant searches among his purchase records for the given data item for one where $halfword\_search_T$ has at least $l/coll\_size$ (half-)symbols in common with $halfword\_search_{red}$.
  - He now tries to extract the value emb_T from the second halfsymbols of $word_{red}$. First he excludes all those symbols $sym_{red,i}$ that definitely do not belong to this traitor, because their first halfsymbols are different from those in $halfword\_search_T$. The remaining second halfsymbols, $halfsym\_emb_{red,i}$, constitute a word with many erasures. The merchant applies the decoding procedure of *EECC* to it and hopes that the result is emb_T.

---

2 Using the index B is only a notational help for us to distinguish the values used with different buyers; of course it does not mean that the merchant has to know this buyer's identity.

5.3 Security of the Construction and Requirements on the Code

We now consider the security of the scheme and find out how many errors the code *EECC* has to tolerate in addition to the erasures. The effectiveness of the scheme, i.e., that embedding yields a reasonable data item for the buyer if nobody cheats, is clear. Recall from the proof sketch of Construction 1 what security requirements we made on a scheme for embedding and extracting:

- Security of the buyer. The merchant should not gain knowledge about emb_B during embedding.
- Security of the merchant. As long as there are at most *coll_size* traitors, extracting will recover the value emb_T used by a traitor with high probability.

The same requirements make the application in a non-anonymous fingerprinting scheme secure.

Security for the buyer. This is clear because the only output the merchant gets from the steps that involve emb_B are commitments and a zero-knowledge proof.

Security for the merchant, overview. First, the properties of the underlying code Γ_0 guarantee that all symbols $sym_{red,i}$, and thus all halfsymbols in $halfword\_search_{red}$, will belong to one of the traitors, with an error probability of at most $2^{-\sigma}$ overall. At least one traitor T^* must therefore have contributed at least $l/coll\_size$ halfsymbols. Thus the merchant's search in the second step of extracting succeeds.

We show in 1. below that for suitably chosen parameters, the merchant almost certainly really identifies the record of a traitor, i.e., no record of an honest buyer fulfils the search criterion.

However, it is not clear that the traitor T whom the merchant identifies contributed at least $l/coll\_size$ entire symbols, nor that all the symbols that he did not contribute will lead to erasures, because different symbols can agree on their first half. But at least we show in 2. below that in a position i where a symbol from a traitor other than T was used, the first halfsymbol is random. Intuitively, this means that the traitors cannot introduce errors instead of erasures on purpose.

Hence there are at most $2^{-\kappa_1}l$ errors on average. We show in 3. below that there are almost always at most $3 \cdot 2^{-\kappa_1}l$ errors. Moreover, the merchant's search criterion immediately implies that there are at most $l - l/coll\_size$ erasures. Hence it is sufficient to use a code *EECC* that tolerates $e = 3 \cdot 2^{-\kappa_1}l$ errors and $r = l - l/coll\_size$ erasures.

Details. We now prove the three statements from the overview and state the necessary constraints on the parameters. As the worst case, we assume that the traitors know their own codewords completely, i.e., they know to which indices the marks they found belong and which version of the data in one mark encodes 0 and 1, respectively.

1. We have to show that almost certainly, no honest buyer's $halfword\_search_B$ will have $l/coll\_size$ symbols in common with $halfword\_search_{red}$. This is a standard proof of collusion-tolerance since [ChFN 94]: The traitors have no information about $halfword\_search_B$, as the merchant is honest in this part of the proof. Hence, when selecting $halfword\_search_{red}$, the probability that they guess a particular halfsymbol of a particular buyer correctly is $p = q_1^{-1}$. Let S be the random variable denoting the number of symbols guessed correctly. By the Chernoff bound, $P(S \geq 3pl) < 2^{-pl}$, i.e., $P(S \geq 3q_1^{-1}l) < 2^{-q_1^{-1}l}$. If we want to bound the overall probability for all N_M buyers by $2^{-\sigma}$, we need $q_1 \geq 3coll\_size$ and $l \geq q_1(\sigma + log_2(N_M))$.
2. We have to show that in every position i where the traitors use a symbol $sym_{red,i} \neq sym_{T,i}$, the equality $halfsym\_search_{red,i} = halfsym\_search_{T,i}$ will only hold with probability $2^{-\kappa_1}$. As the merchant has chosen both these halfsymbols randomly and independently, it suffices to show that

the traitors have no information what values of *halfsym_search* are encrypted by any symbol sym_{red}. We can consider each position i separately, because the merchant does not use any common information in different positions.

The only knowledge the traitors have about the encryption function $subst_i$ is their own symbols $sym_{T^*,i}$ and the corresponding halfsymbols $halfsym\_emb_{T^*,i}$. This is at most as much information as if they knew the precise range of the restricted substitution $subst_i(\bullet, halfsym\_emb)$ for each value of *halfsym_emb*. These substitutions are completely independent random permutations (onto renamed domains). If the attackers select $sym_{red,i} \neq sym_{T,i}$ from the range of $subst_i(\bullet, halfsym\_emb_{T,i})$, then $halfsym\_search_{red,i} = halfsym\_search_{T,i}$ is impossible because of the one-to-one property. Otherwise, they have no information whether the first halfsymbols agree because of the independence of the permutations.

3. Finally, we show that there are almost always at most $3 \cdot 2^{-\kappa_1} l$ errors. We know from 2. that in each position, there is an error with respect to the word of a particular traitor T with probability at most $p = 2^{-\kappa_1} = q_1$. Hence we can use the Chernoff bound as in 1. This leads to the constraint $l \geq q_1(\sigma + log_2(coll\_size))$, if we want to bound the probability by $2^{-\sigma}$ for all traitors together. This constraint is weaker than that in 1.

5.4 Reed-Solomon Codes for Error-and-Erasure Decoding

We first recall the properties of Reed-Solomon codes. All the results mentioned here can be found in [Blah 83]. Reed-Solomon codes are a class of cyclic codes. Any finite field GF(q) can serve as the alphabet; the blocklength is then $l = q - 1$. That the blocklength for a given alphabet is fixed is a certain restriction. For any $t < l/2$, there is a Reed-Solomon code of minimum distance $d = 2t + 1$ and dimension $dim = l - 2t$, and it can be constructed efficiently.[3] This is the maximum dimension possible for the given minimal distance for any linear code; reaching this bound is the main advantage of Reed-Solomon codes.

Usually, a code with minimal distance $d = 2t + 1$ is used to correct up to t errors. However, such a code can also tolerate any combination of e errors and r erasures with $2e + r + 1 \leq d$. This can easily be seen because the restriction of the code to the positions where no erasure occurred still has a minimal distance of at least $d - r$. Furthermore, all BCH codes, of which Reed-Solomon codes are a subclass, can be efficiently decoded for $2e + r + 1 \leq d^*$, where d^* is their so-called designed distance, which equals d for Reed-Solomon codes.

5.5 Setting the Parameters

If we use Reed-Solomon codes in Construction 2, the alphabet size $q_2 = 2^{\kappa_2}$ equals the blocklength l. To tolerate the up to $e = 3 \cdot 2^{-\kappa_1} l$ errors and $r = l - l/coll\_size$ erasures, we need a minimal distance $d = 2t + 1 \geq 2e + r + 1$, which means $2t \geq 6 \cdot 2^{-\kappa_1} l + l - l/coll\_size$. To encode the secrets to be embedded, we need $dim = l - 2t \geq len(k)/\kappa_2 = len(k)/\log_2(l)$. Both inequalities for t can be fulfilled iff l and κ_1 are chosen such that (neglecting rounding errors)

$$-6 \cdot 2^{-\kappa_1} l + l/coll\_size \;\geq\; len(k)/\log_2(l).$$

[3] For concreteness: If α is a primitive element of GF(q), the generator polynomial of this code is $g(x) = (x-\alpha)(x-\alpha^2)\ldots(x-\alpha^{2t})$, i.e., the code consists of the multiples of $g(x)$ by polynomials of degree less than $l-2t$.

Certainly, the left side must be positive; let us require $2^{\kappa_1} \geq 24 coll\_size$. Then l remains to be chosen such that $l \cdot \log_2(l) \geq 4/3 len(k) coll\_size$. Let $l^* := len(k) coll\_size$. One can easily verify that $l \geq 2l^*/\log_2(l^*)$ is a sufficient condition.

6 Conclusion

We have introduced the concept of anonymous fingerprinting, a cryptologic copyright mechanism where honest buyers need not identify themselves to merchants, but merchants can nevertheless find out the identity of traitors who redistribute data without permission. We gave a precise definition of the concept, mentioned some variants, and presented a provably secure framework construction. It can be instantiated with some known schemes for fingerprinting without much collusion tolerance and for collusion-tolerant traitor tracing. To obtain collusion-tolerant fingerprinting, too, we constructed the first collusion-tolerant asymmetric fingerprinting scheme with 2-party trials. Such trials have practical advantages. However, the complexity in the current instantiation with Reed-Solomon codes is somewhat higher than that of known schemes with 3-party trials. A code where the same amount of data could be encoded with a smaller alphabet and a longer blocklength would decrease this problem; however, we are not aware of one where the minimum distance can be very near the blocklength and an efficient procedure for error-and-erasure-decoding is known. Actually, we regard our constructions rather like constructive proofs of existence; however, the additional complexity introduced by anonymity compared with the underlying constructions is not very high.

Acknowledgments

We thank *Matthias Schunter* for interesting discussions and *Rudi Piotraschke* for helpful advice with coding theory.

References

BiMe 96 Ingrid Biehl, Bernd Meyer: Protocols for Collusion-Secure Asymmetric Fingerprinting; accepted for 14th Symposium on Theoretical Aspects of Computer Science (STACS) 1997.

Blah 83 Richard E. Blahut: Theory and Practice of Error Control Codes; Addison-Wesley, Reading 1983.

BlMP 86 G. R. Blakley, C. Meadows, G. B. Purdy: Fingerprinting Long Forgiving Messages; Crypto ’85, LNCS 218, Springer-Verlag, Berlin 1986, 180-189.

BoRD 95 F. M. Boland, J. J. K. Ó Ruanaidh, C. Dautzenberg: Watermarking Digital Images for Copyright Protection; 5th IEE International Conference on Image Processing and its Applications, Edinburgh 1995, 326-330.

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

Bran 94 Stefan Brands: Untraceable Off-line Cash in Wallet with Observers; Crypto ’93, LNCS 773, Springer-Verlag, Berlin 1994, 302-318.

BrCC_88 Gilles Brassard, David Chaum, Claude Crépeau: Minimum Disclosure Proofs of Knowledge; Journal of Computer and System Sciences 37 (1988) 156-189.

BüPf 90 Holger Bürk, Andreas Pfitzmann: Value Exchange Systems Enabling Security and Unobservability; Computers & Security 9/8 (1990) 715-721.

Chau 81 David Chaum: Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms; Communications of the ACM 24/2 (1981) 84-88.

Chau 85 David Chaum: Security without Identification: Transaction Systems to make Big Brother Obsolete; Communications of the ACM 28/10 (1985) 1030-1044.

ChFN 94 Benny Chor, Amos Fiat, Moni Naor: Tracing traitors; Crypto ’94, LNCS 839, Springer-Verlag, Berlin 1994, 257-270.

CKLS 96 Ingemar Cox, Joe Kilian, Tom Leighton, Talal Shamoon: A Secure, Robust Watermark for Multimedia; Information Hiding Workshop, Isaac Newton Institute, University of Cambridge, UK, 1996, Preproceedings, 175-190.

DeYu 91 Yvo Desmedt, Moti Yung: Arbitrated Unconditionally Secure Authentication can be Unconditionally Protected Against Arbiter’s Attacks; Crypto ’90, LNCS 537, Springer-Verlag, Berlin 1991, 177-188.

Pfit 96 Birgit Pfitzmann: Trials of Traced Traitors; Information Hiding Workshop, Isaac Newton Institute, University of Cambridge, UK, 1996, Preproceedings, 43-57.

PfSc 96 Birgit Pfitzmann, Matthias Schunter: Asymmetric Fingerprinting; Eurocrypt ’96, LNCS 1070, Springer-Verlag, Berlin 1996, 84-95.

PfWa 96 Birgit Pfitzmann, Michael Waidner: Asymmetric Fingerprinting for Larger Collusions; accepted for 4th ACM Conference on Computer and Communications Security, 1997; preliminary version IBM Research Report RZ 2857 (#90805) 08/19/96, IBM Research Division, Zurich, 1996.

Wagn 83 Neal R. Wagner: Fingerprinting; 1983 Symposium on Security and Privacy, IEEE, Oakland, California, 18-22.

ZhKo 95 Jian Zhao, Eckhard Koch: Embedding Robust Labels Into Images For Copyright Protection; International Congress on Intellectual Property Rights for Specialized Information, Knowledge and New Technologies, Oldenbourg-Verlag, Vienna 1995.