Communication Optimal Tardos-based Asymmetric Fingerprinting

Aggelos Kiayias[1], Nikos Leonardos[2]*, Helger Lipmaa[3], Kateryna Pavlyk[3], and Qiang Tang[1,4]

[1] National and Kapodistrian University of Athens, Greece
aggelos@di.uoa.gr
[2] Université Paris Diderot – Paris 7, France
nleon@liafa.univ-paris-diderot.fr
[3] University of Tartu, Estonia
helger.lipmaa@gmail.com, kateryna.pavlyk@ut.ee
[4] University of Connecticut, U.S.A
qiang@cse.uconn.edu

Abstract. Asymmetric fingerprinting schemes — introduced by Pfitzmann and Schunter in Eurocrypt 1996 — enable the transmission of a file stored in a server to a set of users so that each user obtains a variation of the file. The security considerations of these schemes are as follows: if any (appropriately bounded) subset of users collude to produce a "pirate" copy of the file, it is always possible for the server to prove to a third party judge the implication of at least one of them, while a malicious server can never implicate innocent users.
Given that asymmetric fingerprinting is supposed to distribute files of substantial size (e.g., media files including video and audio) any communication rate (defined as the size of the file over the total transmission length) less than 1 would render them practically useless. The existence of such schemes is currently open. Building on a rate close to 1 oblivious transfer (constructed from recently proposed rate optimal homomorphic encryption), we present the first asymmetric fingerprinting scheme that is *communication optimal*, i.e., its communication rate is arbitrarily close to 1 (for sufficiently large files) thus resolving this open question. Our scheme is based on Tardos codes, and we prove our scheme secure in an extended formal security model where we also deal with the important but previously unnoticed (in the context of asymmetric fingerprinting) security considerations of *accusation withdrawal* and *adversarial aborts*.

Keywords: Asymmetric Fingerprinting, Tardos Code, Rate Optimal, Group Accusation

1 Introduction

In a fingerprinting scheme, cf. [4], a server (or service provider SP) distributes a file to a set of users. The server has the flexibility to furnish a different version

* Work done while the second author was at University of Athens

of the file to each user. This is done by splitting the file into segments and offering at least two variations per segment. Given these segments, the file can be assembled in a *fingerprinted* fashion: at each segment the variation obtained corresponds to a symbol over an alphabet. Therefore, each user's file determines a string over that alphabet - the user's fingerprint (e.g., the data M is divided into n blocks, for each block i, there are two versions m_i^0, m_i^1, a user assigned with a binary codeword $b_1, \ldots, b_n$ will receive his versions as $m_1^{b_1}|| \ldots ||m_n^{b_n}$). The objective here is that if the users collude to produce a "pirate" version of the file by combining their segments, the server is still capable of discovering (at least some) of the identities of the colluding users.

If the SP alone generates the fingerprints for users and directly transmits them the fingerprinted files, we have what is known as a symmetric fingerprinting scheme. As the server is fully trusted in this setting, the security requirement is that malicious users cannot collude to frame any innocent user or evade the tracing algorithm. The subtle issue in this case is that the server and the user are both able to produce a pirate file so when a pirate copy is brought to light, an honest SP cannot provide an "undeniable" proof that a user is at fault and symmetrically an honest user cannot defend herself against a malicious SP that frames her (say, due to e.g., an insider attack on the SP side).

In order to resolve the above issue, [21] introduced asymmetric fingerprinting schemes in which no one (even the server) should be capable to implicate an innocent user. Thus when a dispute happens, the server can provide a convincing proof that a guilty user is at fault. It follows that the server should not be fully aware of the fingerprint of each user (otherwise it is capable of impersonating them) and hence this suggests that the download of the fingerprinted file should be performed in an oblivious manner from the servers' point of view. Now in this case, the Judge could resolve the dispute between the server and a user (i.e., guilty users will be found guilty by the judge while the server will not be able to implicate an innocent user in the eyes of the judge).

In the original papers [21,22,23] the file transfer stage was treated generically as an instance of secure two party computation. Unfortunately, even with "communication efficient" secure two-party computation [20,8,10,14] the communication overhead of the resulting protocol is prohibitively high (e.g., even with the most communication efficient generic protocols, [10,14], the communication rate — the size of the file over total number of bits transmitted — will be at most 0.5 and their use will impose the additional cost of a prohibitively large CRS which needs to be known a-priori to both client and server). With the discovery of optimal length binary fingerprinting codes by Tardos [26], Charpentier et al. [6] observed that oblivious transfer could be used as a building block for a Tardos-based asymmetric fingerprinting. Their proposed solution however is sub-optimal (it has a rate still at most 0.5) and in order to achieve the fingerprint generation it relies on *commutative encryption*, a primitive not known to be constructible in a way that the resulting scheme can be shown provably secure. Furthermore no complete security analysis is provided in [6] which leaves a number of important security issues unaddressed (specifically "accusation withdrawals" and "selective aborts" – see below).

Achieving rate close to 1 is the most critical open question in the context of asymmetric fingerprinting from an efficiency point of view. Indeed, any asymmetric fingerprinting is particularly sensitive to its communication overhead: the file to be transmitted by the sender can be quite large (e.g., a movie file) and thus any scheme whose communication rate is not close to 1 is likely to be useless in a practical setting. We note that efficient asymmetric fingerprinting schemes can enable more complex applications; e.g., as building blocks for "anonymous buyer-seller watermarking" [25,24]; these systems rely on asymmetric fingerprinting schemes to enable copyright protection (but they do not consider the implementation of such fingerprinting schemes explicitly).

Furthermore, analyzing the security of an asymmetric fingerprinting scheme is involved as the security requirements require that the SP cannot frame an innocent user, while at the same time the malicious users should still not be able to escape from tracing. The analysis should rely both on the security of the protocol and on the property of the code, specifically, no user should be able to produce a pirate file that makes the SP and the judge disagree. Given that Tardos tracing accuses a subset of the users (based on a threshold condition) it is possible for the judge and the SP to disagree on some users. This type of attack has not been considered before; we call it accusation withdrawal as it forces the SP to withdraw an originally made accusation since the judge cannot support it. Ensuring that no accusation withdrawal happens protects the SP from starting accusation procedures that are not going to succeed with high probability. Finally, during the file transfer stage the user may abort. Given that these file transfer procedures can be lengthy (due to the large size of the files to be downloaded) the possibility of an adversary exploiting aborting and restarting as an attack strategy is important to be included in the security model (and in fact we show an explicit attack if many aborts are permitted — see below).

1.1 Our contributions.

Rate-optimality. We propose the first rate-optimal (rate is defined as the size of the actual data over the size of total communication) asymmetric fingerprinting scheme. Our scheme is based on Tardos codes [26]. To achieve this property, we use a rate optimal 1-out-of-2 oblivious transfer ((2,1)-OT), and a new rate-optimal 1-out-of-2 strong conditional oblivious transfer ($(2,1)$-SCOT, [3]). Both are constructed in [16], and they are built on the rate-optimal homomorphic encryption scheme developed in the same paper. Based on these rate optimal protocols, we propose a rate-optimal fingerprinted data transfer protocol (tailored specifically for bias-based codes including Tardos codes).

More precisely, in a fingerprinted data transfer protocol, the sender has as private input two pairs of messages and biases. The sender and the receiver simulate two private biased coin tosses using SCOT and the receiver obtains one message from each pair (which one of the two it receives, is determined by the outcome of the biased coin flip). The actual message transmission is based on the rate optimal OT protocol. Furthermore the sender selects randomly one of the two SCOT-s and revokes its receiver security, i.e., the sender will learn which one of the two versions the receiver has obtained in this SCOT. This partial revocation of receiver-security will enable the sender to correlate "pirate" files that are generated by coalitions of malicious users. Our final scheme inherits the communication efficiency of the underlying SCOT and OT protocols and thus it is communication-optimal: the rate of each data transfer is asymptotically 1.

A complete security analysis in the (extended) Pfitzmann-Schunter model: we analyze the security of our construction in an *extended* version of the Pfitzmann-Schunder model [22]. The extension we present is two-fold: first we extend the model to capture the setting of multiple accusations. In the original modeling only a single colluder was required to be accused. In the extended model we allow a set of users to be accused. This accommodates accusation algorithms that are based on Tardos fingerprinting [26] that have this capability. Group accusation in asymmetric schemes needs special care from a security point of view: it makes the system prone to *accusation withdrawal*, the setting where the server will have to withdraw an accusation because the judge is unable to verify it. We demonstrate (through actual implementation experiments, see Fig 2) that the straightforward application of Tardos identification (as may naively be inferred from the description of [6]) does not preclude accusation withdrawal. We subsequently show how to modify the accusation algorithm between judge and server so that no accusation withdrawal can take place. Our second model extension concerns the explicit treatment of the abort operation within the security model: all known asymmetric fingerprinting schemes rely on two-party coin tossing. Given that *fair* coin tossing is known to be unattainable [7] it follows that it may be possible for an adversarial set of users to exploit this weakness and utilize a transmission abort strategy with the purpose of evading detection. We demonstrate that an explicit treatment of this in the security model is essential as if one enables users to restart after an abort, it is possible to completely break server security! (this fact went entirely unnoticed before). By properly controlling aborts and restarts we show how security can be maintained.

2 Rate-Optimal OT and SCOT Protocols

We recall that an oblivious transfer (OT) protocol and a strong conditional oblivious transfer (SCOT, [3]) protocol for predicate Q (s.t. $Q(x, y) \in \{0, 1\}$) implement securely the following functionalities respectively (W.l.o.g., assume $|m_0| = |m_1|$):

$$f_{\mathsf{OT}}(b, (m_0, m_1)) = (m_b, \perp),\ f_{Q-\mathsf{SCOT}}(x, (y, m_0, m_1)) = (m_{Q(x,y)}, \perp) \ .$$

Here, we will use the rate optimal OT and SCOT protocols derived in [16] from their recently developed rate optimal large-output branching program homomorphic encryption (LHE) scheme. We recall that their LHE scheme enables the receiver to compute on ciphertexts the value $f(x, y)$, where x is his input,

y is sender input, and f is an arbitrary function that can be evaluated by a polynomial-size (integer-valued) branching program. In the LHE scheme of [16], the receiver encrypts (by using a variant of the Damgård-Jurik cryptosystem [9]) his input x, and sends the ciphertext $\mathsf{Enc}_r(x)$ to the sender. The sender evaluates privately large-output branching programs like in [15,19], but does it in a communication-preserving manner. Let the output of the evaluation be denoted as $\mathsf{Eval}(P, \mathsf{Enc}_r(x))$, where P is a large-output branching program that evaluates $f(\cdot, y)$ on input x. The sender returns a single "ciphertext" to the receiver, who then (multiple-)decrypts it as in [15,19]. The rate of the LHE scheme is defined as $r = (|x| + |P(x)|)/(|\mathsf{Enc}_r(x)| + |\mathsf{Enc}_r(P(x))|)$. Assuming $|f(x, y)|$ is large, [16] showed by using an intricate analysis how to achieve a rate $1 - o(1)$. We refer to [16] for more information.

Rate-optimal OT. As shown in [16], one can define a rate-optimal $(2, 1)$-oblivious transfer protocol as follows. Let the server have a database (m_0, m_1) and assume that $P[x, (m_0, m_1)] = m_x$ for $x \in \{0, 1\}$. Thus, the size of P is 1. Since rate-optimal $(2, 1)$-OT has many applications, we will call it *oblivious download* (OD). Let $\mathsf{OD}_s[\mathsf{Enc}_r(x), (m_0, m_1)]$ denote the server side computation in this protocol, given client ciphertext $\mathsf{Enc}_r(x)$ and server input (m_0, m_1).

Rate-optimal SCOT. Also, as shown in [16], one can use the LHE of to construct an efficient SCOT protocol for the functionality $f_{Q-\mathsf{SCOT}}(x, (y, m_0, m_1))$, where Q has a polynomial-size branching program (i.e., $Q \in \mathbf{L/poly}$), as follows. Let P' be an efficient branching program that evaluates the predicate $Q(x, y)$. Let P be a large-value branching program, obtained from P' by just replacing the leaf value 0 with m_0 and 1 with m_1. The LHE scheme (and thus also the resulting SCOT protocol) will have computation, linear in the size of P', and communication $(1 + o(1))(|x| + |m_0|)$ and thus rate $1 - o(1)$. In the rest of the paper we will need the next instantiation of a new rate-optimal SCOT protocol.

Rate-optimal SCOT for the LEQ predicate. Denote $\mathsf{LEQ}(x, y) := [x \leq y]$. It is easy to see that LEQ can be evaluated by a branching program of size and length $\ell := \max(|x|, |y|)$. Thus, one can implement $f_{\mathsf{LEQ-SCOT}}$ securely in time $\Theta(\ell)$ and rate $1 - o(1)$. Let us denote server computation in this protocol as $\mathsf{LEQ}_s[\mathsf{Enc}_r(x), (y, m_0, m_1)]$.

Remark. The security of the $\mathsf{OD}, \mathsf{SCOT}$ protocols are simple corollaries of the security proofs from [15,16]. Also, one can also use an arbitrary efficient — with communication $o(|m_i|)$ — millionaire's protocol, like the one in [12,3,18] to find out $b = [x < y]$, and then use the oblivious download protocol to implement an optimal-rate SCOT protocol for the LEQ predicate. However, we think that the use of optimal-rate LHE from [16] (instead of composing a millionaire's protocol and an OD protocol) is more elegant.

3 Fingerprinted Data Transfer for Bias-Based Codes

In this section, we will introduce the main building block of our Tardos-based asymmetric fingerprinting scheme, which we call fingerprinted data transfer.

As our fingerprinting scheme relies on the properties of fingerprinting codes (we only focus on binary codes here), let us first recall the basics about fingerprinting codes. A *binary fingerprinting code* [17] is a pair of algorithms $(\mathsf{gen}, \mathsf{trace})$, where gen is a probabilistic algorithm taking a number N, an optional number (upper-bound on the detected coalition size) $t \in [N] = \{1, \ldots, N\}$ and security parameter ϵ as input and outputs N bit-strings $\mathcal{C} = \{C_1, \ldots, C_N\}$ (called codewords), where $C_i = \mathsf{c}_1^i \ldots \mathsf{c}_n^i$ for $i \in [N]$ and a tracing key tk. trace is a deterministic algorithm inputting the tracing key tk and a "pirate" codeword C^*, and outputting a subset $\mathcal{U}_{acc} \subseteq [N]$ of accused users. A code is called *bias-based* [2] if each codeword $C_j = \mathsf{c}_1^j \ldots \mathsf{c}_n^j$ is sampled according to a vector of biases $\langle p_1, \ldots, p_n \rangle$, where $\forall j \in [N]\, \forall i \in [n], \Pr[\mathsf{c}_i^j = 1] = p_i$, and $p_i \in [0, 1]$.

A fingerprinting code is called $t-$*collusion resistant* (*fully collusion resistant* if $t = N$) if for any adversary $\mathcal{A}$ who corrupts up to t users (whose indices form a set $\mathcal{U}_{cor} \subset \{1, \cdots, N\}$), and outputs a pirate codeword $C^* = \mathsf{c}_1^* \ldots \mathsf{c}_n^*$ (which satisfies the marking assumption, i.e., for each $i \in [n], \mathsf{c}_i^* = \mathsf{c}_i^j$ for some $j \in \mathcal{U}_{cor}$), $\Pr[\mathcal{U}_{acc} = \emptyset \text{ or } \mathcal{U}_{acc} \not\subseteq \mathcal{U}_{cor} : \mathcal{U}_{acc} \leftarrow \mathsf{trace}(tk, C^*)] \leq \epsilon$ (i.e., the probability that no users are accused or an innocent user is accused is bounded by ε).

We also recall the Tardos code [26] $F_{nt\epsilon}$ here, it has length $n = 100t^2k$, with $k = \log \frac{1}{\epsilon}$. The gen algorithm generates a codeword as follows. For each segment index $j \in [n]$, it chooses a bias $p_j \in [0, 1]$ according to a distribution μ (see [26] for the definition of μ). Each bias satisfies $\frac{1}{300t} \leq p_j \leq 1 - \frac{1}{300t}$, where t is the collusion size. For each codeword $C = \mathsf{c}_1 \ldots \mathsf{c}_n$ outputted by gen, $\Pr[\mathsf{c}_j = 1] = p_j$, and $\Pr[\mathsf{c}_j = 0] = 1 - p_j$ for all $j \in [n]$. Regarding security, there is a trace algorithm such that, for any coalition of size at most t, with probability at least $1 - \epsilon^{t/4}$ accuses a member of the coalition, while any non-member is accused with probability at most ϵ.

3.1 Definitions of fingerprinted data transfer

Now we define our main building block of fingerprinted data transfer (FDT for short). Recall that each user should receive a fingerprinted copy of the file according to his codeword. In the case of asymmetric fingerprinting, the segments of the file will be transferred in an oblivious fashion so that the server should be aware of only half of the user fingerprinting code. To be more specific, all segments are transmitted using oblivious transfer to enable the user to receive one of the versions, and for each pair of segments $(2i - 1, 2i)$, where $i \in [n]$, the server will know one of the segments, the version that the user receives.

Intuitively, if we double the length of the fingerprinting code (dividing the file into $2n$ segments), each user is essentially assigned two codewords, one is known to the server, thus the trace algorithm can be executed to identify malicious users; the other one is unknown to the server, and will be revealed to the judge only when dispute happens. A user will be accused only when both codewords are considered contributing to a pirate file. In this way, a malicious SP $\mathcal{S}$ frames an honest user unless innocent users may be accused in the fingerprinting code.

We also need to be careful that if malicious users know which half of the codeword is known to the server, they may collude in a way that every codeword in the collusion only contribute to half of the file thus no one will be accused on both fingerprints. Thus for the segments $(2i-1, 2i)$ for $i \in [n]$, the index that the segment version is revealed to the server is also oblivious to the user.

The asymmetric fingerprinting scheme will essentially be running FDT (defined below) in parallel for all pairs of the segments $(2i-1, 2i)$, thus it is enough for us to illustrate the idea by considering only the sub-protocol for one pair of segments. As Tardos code is binary, there are only two versions for each segment. Consider two parties, a sender $\mathcal{S}$ and a receiver $\mathcal{R}$. The sender has two pairs of messages, two rational valued "biases" in $[0,1]$ and one bit c as inputs. The receiver has no input. After the execution of the FDT protocol, $\mathcal{R}$ will sample one message from each of the two pairs following the binary probability distribution defined by the two biases and $\mathcal{S}$ will learn the output of the receiver for the c-th pair. This describes the ideal operation of the primitive for the case of one pair of segments. It is straightforward to extend to an arbitrary number of pairs. The following is the formal definition of the fingerprinted data transfer for bias-based codes including our main target Tardos code [26]. And following the standard simulation base paradigm [13], we can also define the security of the FDT protocol, and we defer it to the full version.

Definition 1. *A* fingerprinted data transfer functionality *Π involves two parties, a sender $\mathcal{S}$ and a receiver $\mathcal{R}$. The sender inputs two biases $p_0, p_1 \in [0,1]$, four messages $(m_0^0, m_0^1), (m_1^0, m_1^1)$, and a bit $c \in \{0,1\}$; at the end of the protocol, $\mathcal{R}$ outputs $\{m_i^{b_i}\}$ for $i, b_i \in \{0,1\}$ such that $\Pr[b_i = 1] = p_i$; while $\mathcal{S}$ outputs b_c. We can express this (probabilistic) functionality as:*

$$\Pi[\perp, ((p_0, p_1), (m_0^0, m_0^1, m_1^0, m_1^1), c)] = [(m_0^{b_0}, m_1^{b_1}), b_c], \textit{where } \Pr[b_i = 1] = p_i$$

Somewhat similar functionalities have been used for completely different applications, see, e.g. [11,1]. The FDT protocol of Sect. 3.2 might be modified so as to be used in these applications; we omit further discussions.

3.2 A Communication-Optimal Fingerprinted Data Transfer Protocol

On the receiver $\mathcal{R}$ side, for each pair of messages, say pair 0, FDT will enable an oblivious sampling of one message from (m_0^0, m_0^1) according to the bias p_0, i.e., $\mathcal{R}$ receives m_0^1 with probability p_0. To enable efficient oblivious sampling, suppose $p_0 \approx t_0/T$ for some t_0, where $T = 2^\ell$ and ℓ is the precision level (this provides an exponentially good approximation). To run a coin tossing protocol to generate a random coin u, $\mathcal{R}$ and the SP $\mathcal{S}$ can utilize a SCOT protocol (e.g., [3]) to transmit the data in a way that the user receives m_0^1 iff $u \leq t_0$. Doing this will allow the receiver to get m_0^1 with probability close to $p_0 = t_0/T$. Furthermore, they can run such procedure twice for the two pairs, and then run a $(2,1)$-OT protocol to reveal one of the bit to the SP.

Unfortunately, directly applying the SCOT protocol from, e.g, [3] will result in a communication rate as low as $1/\ell$, as the sender has to send ℓ ciphertexts with similar size to m_i^0. Moreover, a malicious user may abort after receiving the file without revealing half of his bits. To deal with these concerns, our protocol will be divided into two phases, the first (the handshake phase) samples only the codewords according to the biases that are specified by the sender; the second (the content-transfer phase) transfers the actual content according to the codewords that have been drawn. In our implementation, we will only use the SCOT protocol to sample the distribution (transfer only short messages) and then employ a rate-optimal OT protocol (we call oblivious download, OD for short)to execute the content-transfer after the OT protocol is run in which the SP is the receiver and the SP sees one of the bits. We assume that during the hand-shake phase, the sender and receiver exchange their public keys with the corresponding certificates.

Now we proceed to construct the new fingerprinted data transfer protocol. Suppose the sender has $p_0 = \frac{t_0}{T}$, $p_1 = \frac{t_1}{T}$; these determine two distributions over $\{0,1\}$. The sender also has two pairs of messages as inputs $(m_0^0, m_0^1), (m_1^0, m_1^1)$, and prepares another two pairs of $(h_0^0, h_0^1), (h_1^0, h_1^1)$, where $h_i^b = H(m_i^b)|i|b$ for $i, b \in \{0,1\}$. We assume that H is a collision resistant hash function shared by the sender and the receiver, and Com is a secure (binding and hiding) commitment scheme. We choose Enc_r and Enc_s to be good rate additive homomorphic encryption schemes (e.g. using the Damgard-Jurik [9] encryption to encrypt the message bit by bit as in [16]). Here, $\mathcal{R}$ knows the secret key of Enc_r and $\mathcal{S}$ knows the secret key of Enc_s. Recall that $\mathsf{LEQ}_s[\mathsf{Enc}_r(x), (y, m_0, m_1)]$ denotes the sender computation in a concrete SCOT protocol that implements $f_{\mathsf{LEQ-SCOT}}$, and $\mathsf{OD}_s[\mathsf{Enc}_r(x), (m_0, m_1)]$ denote the computation of the server in this protocol, given client ciphertext $\mathsf{Enc}_r(x)$ and server input (m_0, m_1) (defined in Section 2). The full protocol of FDT is presented in Fig 1.

We can estimate the communication rate α of our FDT protocol as follows:

$$\frac{1}{\alpha} \approx \frac{2(|\mathsf{Com}(r_0)| + |\mathsf{Enc}_r(s_0)| + |\mathsf{Enc}_r(h_0^{b_0})|) + |\mathsf{Enc}_s(c)| + |\mathsf{Enc}_s(h_c^{b_c})| + 2|\mathsf{Enc}_r(m_0^{b_0})|}{2|m_0^{b_0}| + 1}$$

$$\approx \frac{o(|m_0^{b_0}|)}{2|m_0^{b_0}|} + \frac{|\mathsf{Enc}_r(b_0)| + |\mathsf{Enc}_r(m_0^b)|}{|b_0| + |P[b_0, (m_0^0, m_0^1)]|} \to \frac{1}{r}, \text{when } m \to \infty \ ,$$

where m is the message size and r is the rate as defined in Sect. 2. We can group several terms into $o(|m_0^{b_0}|)$ as all those are encryptions (or commitments) of fixed size short messages. Thus, when the LHE scheme is rate optimal as [16], our FDT protocol (and further our asymmetric fingerprinting scheme, see next section) is also rate optimal.

Security analysis. We briefly explain the properties of our protocol in the semi-honest model. Correctness follows from the coin tossing and the property of the LHE [16]. For instance, $\overline{\mu_0} = \mathsf{Enc}_r(h_0^1), C_0 = \mathsf{Enc}_r(m_0^1)$, if $r_0 \oplus s_0 \le t_0$, in this case, $\Pr[b_0 = 1] = t_0/T = p_0$. For security, as we are working in the semi-honest model for now, the sender and receiver views can be simulated easily to preserve the consistency with the output. For detailed proofs, we refer to the full version.

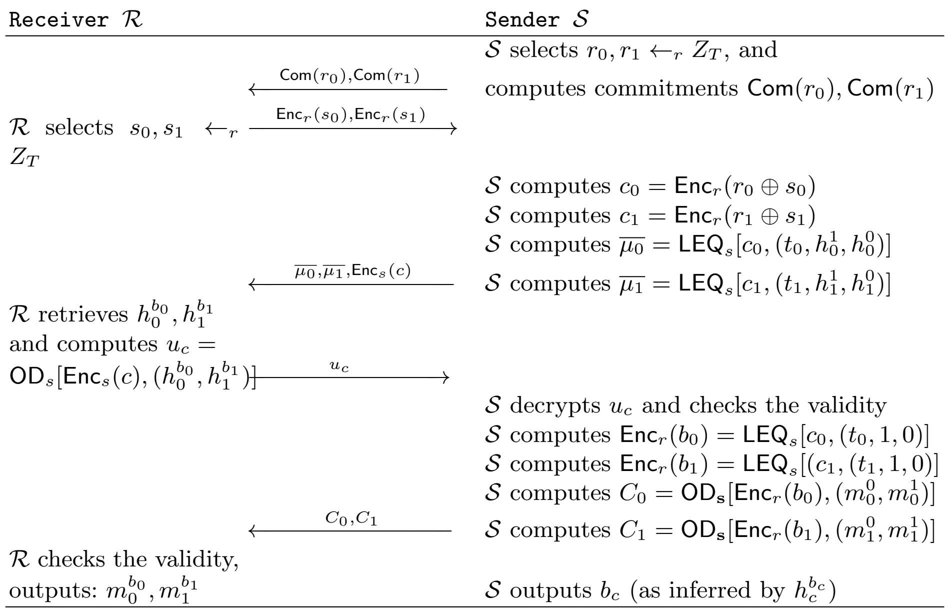
Receiver $\mathcal{R}$
Sender $\mathcal{S}$
$\mathcal{S}$ selects $r_0, r_1 \leftarrow_r Z_T$, and
computes commitments $\mathsf{Com}(r_0), \mathsf{Com}(r_1)$
$\xleftarrow{\mathsf{Com}(r_0),\mathsf{Com}(r_1)}$
$\mathcal{R}$ selects $s_0, s_1 \leftarrow_r Z_T$
$\xrightarrow{\mathsf{Enc}_r(s_0),\mathsf{Enc}_r(s_1)}$
$\mathcal{S}$ computes $c_0 = \mathsf{Enc}_r(r_0 \oplus s_0)$
$\mathcal{S}$ computes $c_1 = \mathsf{Enc}_r(r_1 \oplus s_1)$
$\mathcal{S}$ computes $\overline{\mu_0} = \mathsf{LEQ}_s[c_0, (t_0, h_0^1, h_0^0)]$
$\mathcal{S}$ computes $\overline{\mu_1} = \mathsf{LEQ}_s[c_1, (t_1, h_1^1, h_1^0)]$
$\xleftarrow{\overline{\mu_0},\overline{\mu_1},\mathsf{Enc}_s(c)}$
$\mathcal{R}$ retrieves $h_0^{b_0}, h_1^{b_1}$
and computes $u_c = \mathsf{OD}_s[\mathsf{Enc}_s(c), (h_0^{b_0}, h_1^{b_1})]$
$\xrightarrow{u_c}$
$\mathcal{S}$ decrypts u_c and checks the validity
$\mathcal{S}$ computes $\mathsf{Enc}_r(b_0) = \mathsf{LEQ}_s[c_0, (t_0, 1, 0)]$
$\mathcal{S}$ computes $\mathsf{Enc}_r(b_1) = \mathsf{LEQ}_s[(c_1, (t_1, 1, 0)]$
$\mathcal{S}$ computes $C_0 = \mathsf{OD}_\mathbf{s}[\mathsf{Enc}_r(b_0), (m_0^0, m_0^1)]$
$\mathcal{S}$ computes $C_1 = \mathsf{OD}_\mathbf{s}[\mathsf{Enc}_r(b_1), (m_1^0, m_1^1)]$
$\xleftarrow{C_0,C_1}$
$\mathcal{R}$ checks the validity,
outputs: $m_0^{b_0}, m_1^{b_1}$
$\mathcal{S}$ outputs b_c (as inferred by $h_c^{b_c}$)

Fig. 1. Fingerprinted Data Transfer. $\{(m_b^0, m_b^1), p_b = \frac{t_b}{T})\}_{b=0,1}, c$ are inputs of $\mathcal{S}$

Lemma 1. *Our protocol shown in Fig. 1 securely implements the fingerprinted data transfer functionality. Specifically, it is correct; and it satisfies receiver security if the underlying encryption* Enc_r *is IND-CPA secure; it satisfies sender security if the underlying commitment scheme* $\mathsf{Com}(\cdot)$ *is computationally hiding, and the encryption* Enc_s *is IND-CPA secure.*

Work in the malicious model. Ultimately, we would like to design protocols to defend against malicious adversaries who may arbitrarily deviate from the protocol. The general method that in every step, both parties deploy zero-knowledge proofs to show that they follow the protocol, could be inefficient. Note that our protocol is highly structured, user misbehaviors can be easily detected by the SP with some routine checks about the consistency in the transcripts. In the 2nd round in coin tossing phase, the user could not learn any extra information by not following protocol, as simulation for malicious user is not influenced by the choices of s_0, s_1. While in the 4th round, the SP checks the validity of $h_c^{b_c} = H(m_c^{b_c})||c||b_c$, if u_c is not calculated as in the protocol and passes the checking, it means the malicious user finds a value equal to $H(m_c^{1-b_c})$. As the message segment $m_c^{1-b_c}$ has sufficient entropy thus $h_c^{1-b_c}$ is also unpredictable, otherwise, the user could easily find a collision by randomly sample messages from the distribution of $m_c^{1-b_c}$. To be more specific, suppose M is the space of $m_c^{1-b_c}$ and $\mathcal{D}$ is its distribution, and $H(M) = \{H(m) : m \in M\}$, $\mathcal{D}$ will induce a distribution $\mathcal{H}$ on $H(M)$. Suppose the sender can predict $h(m_c^{1-b_c})$ with probability δ, then the maximum probability of $\mathcal{H}$ is no less than δ. Let us use h_0 to denote the most probable value in $H(M)$. The adversary $\mathcal{A}$ simply sample m_i randomly according to $\mathcal{D}$, and computes the hash value. Following the Chernoff bound, using $O(\frac{1}{\delta^2})$ many samples, $\mathcal{A}$ will almost certainly reach h_0 twice. At the same time, the probability that there are two same messages appear in the sampled messages is exponentially small, as the most probable message from $\mathcal{D}$ appears with negligible probability. Based on these two facts, $\mathcal{A}$ found a collision.

Regarding malicious SP, the user can also do some simple checks of the consistency of the hash values. Note that there is a trusted judge that makes the final decision about the set of accused users. We will show in next section (as the judge is not involved with the FDT protocol) how we can take advantage of this third-party to "force" the SP to follow the protocol, by adding some simple and efficient "proofs of behavior". We require the SP signs on each round of messages she sends together with the user identity, and the user also signs on each round of messages he sends. We also let user store part of the transcripts and reveal them to the judge in case of dispute. Through a careful analysis of Tardos code property together with these simple mechanisms, we can argue security of our asymmetric fingerprinting scheme in the malicious model.

4 An Optimal Asymmetric Fingerprinting Scheme Based on Tardos Code

Pfitzmann and Schunter [22] define an asymmetric fingerprinting scheme to be a collection of four protocols $\langle\mathsf{key\_gen}, \mathsf{fing}, \mathsf{identify}, \mathsf{dispute}\rangle$. The algorithm $\mathsf{key\_gen}$ can be used by a user to produce a public and a secret-key. The protocol fing is a two-party protocol between a user and an SP that will result in the user obtaining the fingerprinted copy of the file and the SP receiving some state of the user codeword. The algorithm $\mathsf{identify}$ is an algorithm that, given a pirate copy and the state of the SP, outputs a non-empty set of public keys (corresponding to the accused user identities). Finally the algorithm $\mathsf{dispute}$ is a 3-party protocol between the judge (or arbiter as it is called in [22]), the user and the SP that either accepts the SP's accusation or rejects it (depending on the evidence presented by the involved parties). For brevity we refer to [22] for the full syntax of the scheme. Regarding the security model, an asymmetric fingerprinting scheme should satisfy two security properties: (i) *security for the SP*, that states that no malicious coalition of less than t users can escape the accusation of one of its members from the $\mathsf{identify}$ algorithm as well as the validation of this accusation by the $\mathsf{dispute}$ protocol, and (ii) *security for the user*, that states that an innocent user cannot be implicated by a malicious SP (who can also corrupt other users) in being responsible for a certain pirate copy. For formal definitions of an asymmetric fingerprinting scheme, we refer to the full version.

In addition to the above basic requirements, we put forth two additional properties that will be of interest.

Communication efficiency. The communication rate of an asymmetric fingerprinting scheme is measured as the ratio of the length of the file that is distributed to the users over the total communication complexity of the fing protocol. In a communication optimal asymmetric fingerprinting scheme it holds that the rate approaches 1 as the length of the file becomes larger. All known schemes in the literature [21,22,23,6] have rate at most 0.5.

Security for the SP under group accusations. In [22] the algorithm identify is responsible for producing a single colluder whose implication is guaranteed to be validated by the dispute algorithm. In [6] this is extended to group accusation, i.e., the identify algorithm produces a set of accused users as output (this is possible given that the underlying fingerprinting code enables such group accusation). For SP security to be preserved under group accusations however, it should be the case that for each accused user, its implication to the construction of the pirate copy is validated by the dispute protocol. In the other case, the SP will have to withdraw at least one accusation (something that may lead to problems in a practical deployment). Therefore a protocol solution should guarantee in the setting of group accusation no accusation withdrawal can occur with non-negligible probability. We refer the formal definitions to the full version.

4.1 Our construction.

We next describe our construction which satisfies the original security requirements of [22] as well as the two properties that we described above. Specifically it is the first asymmetric fingerprinting scheme with both optimal communication complexity and code length. And one can easily adapt our construction to other asymmetric fingerprinting scheme from any bias-based code.

Recall the definition of Tardos code as explained in section 3, the main task is the fing protocol, which will be constructed from our fingerprinted data transfer protocol (see Fig 1) with some extra checks to achieve security in the malicious model in which the adversary may not follow the protocol. To describe the generation of the fingerprinted copy of each user in more detail, let us abstract each variant of a segment m_i^b with a value in $\{0,1\}$,where $i \in [2n], b \in \{0,1\}$ and $2n$ is the length of the fingerprint. Thus, the fingerprinted file of each user is a $2n$-bit string, where each bit signifies which variant of the corresponding segment the user received. It will be generated so that n bits from a set $L \subseteq [2n]$ will be known to the SP, while the other n bits (from $[2n] \setminus L$) will only be known by the user. The user, however, will not know if a given location belongs to L or not. Each of the parts L and $[2n] \setminus L$ is an instance of the Tardos code [26]. The two parts are produced by generating two segments at a time, using the functionality achieved by the protocol in Figure 1, i.e., for the i-th pair of segments $(2i-1, 2i)$, where $i \in [n]$, the user and the server runs the fingerprinted data transfer with the SP taking $[(p_{2i-1}, p_{2i}), (m_{2i-1}^0, m_{2i-1}^1), (m_{2i}^0, m_{2i}^1), c_i]$ as inputs. Based on the security properties of this protocol, the user receives two variants of two different segments, while the SP does not know one of them and the user is oblivious regarding which one the SP knows. Our asymmetric fingerprinting scheme proceeds as follows:

Key generation. The key_gen for the user is simply the generation of two public-secret key pairs $(pk_1, sk_1), (pk_2, sk_2)$. The first is for a digital signature scheme (which we use as black-box), and the second is for the additively homomorphic encryption Enc_r used in the fingerprinted data transfer.

The fing protocol. The user has as input its public and secret keys while the SP has as input SP public keys, and system parameters, e.g., the level of precision ℓ. Furthermore, the protocol is stateful from the point of view of the SP. The state of the SP contains the definition of the Tardos code parameters (e.g., probabilities $\{p_i\}$). Also, the SP has as private inputs a set $L = \{c_1, \ldots, c_n\}$, and a file that is divided in $2n$ segments for each one of which there are two variations. The i-th segment, b-th variant is denoted by m_i^b.

The fing protocol proceeds as follows: the SP and the user first carry out a handshake protocol to prepare the system parameters including the exchange of the public keys of each other; then for each i-th pair of segments with indices $(2i-1, 2i)$ where $i \in [n]$, the user and the server runs the FDT with the SP taking $[(p_{2i-1}, p_{2i}), (m^0_{2i-1}, m^1_{2i-1}), (m^0_{2i}, m^1_{2i}), c_i]$ as inputs, and these n protocols are run in parallel. Also in each round, if the SP sends out a message, she signs on the message together with the user's identity; if the user sends a message, he signs it as well. During protocol execution each party verifies that the messages they receive are proper and if they are not they will abort the protocol.

Furthermore some additional checks are in place to detect the malicious behavior within fing as explained at the end of section 3.2. These are as follows: The user checks after receiving the actual data segments (in the last round) whether they are consistent with the hash values (see Remark in section 3.2) he received in the 3-rd round. The SP, checks the validity of the hash value she received in the 4-th round. Also, both parties will store some information for input to the $\mathsf{dispute}$ protocol. The user keeps the commitments received from the first round and the hashed values and the encrypted bits of $\{\mathsf{Enc}_s c_i\}$ for $i \in L$, received in the 4-th round; the SP keeps the encrypted random coins of the user received in the 2nd round. Note that these checks do not enforce semi-honest behavior - nevertheless we will show (see Theorem 1,2) they are sufficient for security against malicious parties in the presence of the judge (which is assumed honest).

We see that our fing protocol essentially runs out FDT protocol in parallel with only some extra signatures, thus it inherits the rate optimality from FDT.

The $\mathsf{identify}$ algorithm. This algorithm takes a pirate file M, and all users' half codeword $X_1, \ldots, X_N$ together with the location indices $L_1, \ldots, L_N$ and the vector of biases $\langle p_1, \ldots, p_{2n}\rangle$ as inputs. It first extracts a codeword $Y = y_1 \ldots y_{2n} \in \{0,1\}^{2n}$ from M (as we assume each bit is publicly computable from the segment). For the ease of presentation, we describe the algorithm for one user with stored codeword $X = x_{c_1} \ldots x_{c_n}$ and $L = \{c_1, \ldots, c_n\}$. For each $j \in L$, it computes:

$$U_j = \begin{cases} \sqrt{\frac{1-p_j}{p_j}}, & \text{if } x_j = 1; \\ -\sqrt{\frac{p_j}{1-p_j}}, & \text{if } x_j = 0 \end{cases}$$

as in [26]. The service provider calculates the score of the user over the locations in L; $S = \sum_{j \in L} y_j U_j$, and if $S > Z$, where $Z = 20tk$, the SP reports this user to the judge. This is repeated for every user and a list of accused users is compiled and reported to the judge.

The dispute protocol. This is a protocol among the SP, the judge and a user. The two parties first submit to the judge the protocol transcript they stored. In more detail, the judge requires the user to submit SP's commitments sent in the 1st round and the hash values; also, the judge requires the SP to submit the biases and the encryptions from the user in the 2nd round as well as openings of her commitments. The judge first verifies the code parameters, then does the following checks, (1). the validity of the segments, i.e., they should be one of the versions. (2). the validity of all signatures, if any signature is invalid, accuse the party who submits it. (3). Otherwise, the judge checks whether the user codeword is generated properly, i.e., each bit of the codeword is consistent with the coin-tosses – whether $b_i = [r_i + s_i \leq t_i]$ where b_i is the i-th bit, r_i, s_i, t_i are as in the FDT (the notation $[\cdot]$ here denotes a predicate). To finish this check, the judge requires the SP to open her commitments, and the user to reveal his coins in the ciphertext $\{\mathsf{Enc}_r(s_i)\}$and prove their validity. (4). Furthermore, the judge requests from the user to submit the encrypted locations $\{\mathsf{Enc}_s(c_i)\}$ and requests the SP to decrypt it and prove a correct decryption, so that the judge calculates the set of locations L. Any party failed to prove the correct decryption will be accused.

If all the checks pass, the judge will recover user's fingerprint x' from the bits $\{b_i\}$, also he inspects the pirate content and extracts the fingerprint y'. Then he computes the U' as in the *identify* algorithm for locations $L' = [2n] \backslash L$ using x', y' as inputs. Finally, for any user reported, the judge calculates his score over the locations in L'; $S' = \sum_{j \in L'} y'_j U'_j$, and make decisions if $S' > Z'$, where $Z' = Z/2 = 10tk$, he validates the accusation; otherwise, the user is acquitted.

Note that we are using a lower threshold on the judge-side, to counter-balance the probability that a user is accused over L, but not over $[2n] \setminus L$. In fact this is an essential feature of our scheme to ensure security for the SP under group accusations. We in fact show that if $Z' = Z$ it can happen with high probability that the SP will have to withdraw an accusation; in Fig 2, we explore this experimentally by having a coalition of 40 users where the pirate strategy is as follows: the pirate content is formed via a majority strategy by the segments available to the coalition of size t. For each segment with probability p (a parameter of the strategy) the pirate segment is determined with probability p to be the majority of the segments of all t users or with probability $1 - p$ the segment of the first user. We variate the parameter p of the strategy and we demonstrate from experimental data that for suitable choice of p the number of accusation withdrawals can be as high as as a quarter of the coalition. One would expect that in practice, such high level of accusation withdrawal would impact seriously the credibility of the SP. In our construction, by appropriately differentiating the tracing algorithm between the judge and the SP we circumvent this problem entirely. It should be noted that this issue was not addressed in [6] where the Tardos tracing algorithm was also used for determining the implication of users.

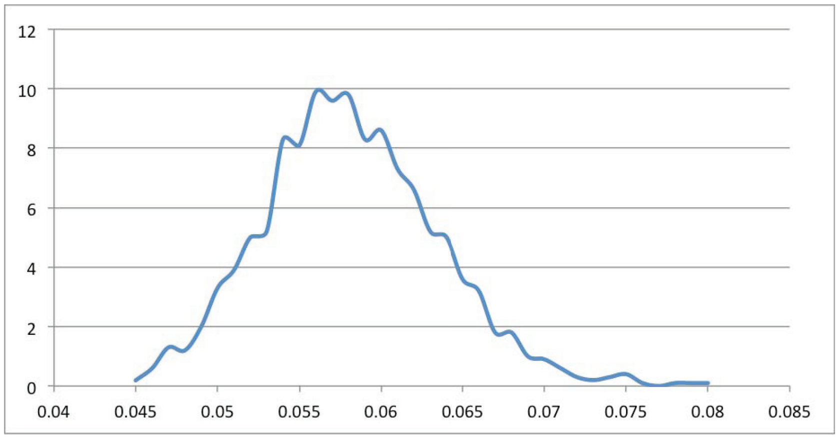

Fig. 2. The vertical axis represents the number of accusation withdrawals (i.e., for how many users the service provider has to abandon its accusation); the total number of colluding users is 40. The horizontal access is the parameter of the colluding strategy p; for a suitably choice of p the accusation withdrawals reach 25% of colluders.

Remark: There are two phases when the judge requests one of the two parties to prove a valid decryption of a bit. As we are using a variant of DJ encryption [9], the prover can simply reveal the message together with the random coins used in encryption as it decodes uniquely in this form. Specifically, if a message m is encrypted with random coin r, it results in a ciphertext $c = (n+1)^m r^{n^s} \mod n^{s+1}$, the prover can decrypts c to recover m and then obtains $r = d^{n^{-s} \mod \phi(n)} \mod n$, where $d = r^{n^s} \mod n$ is computed from $c \cdot (n+1)^{-m} \mod (n^{s+1})$.

4.2 Security analysis.

We give here explain the intuitions about the security analysis of our asymmetric fingerprinting scheme, for the details of the proofs, we refer to the full version.

Security for the innocent user. We will show that no innocent user will be framed. In the case of a semi-honest SP, she follows the fingerprinted data transfer protocol and the accusation procedure, but will try to make a pirate copy to frame some innocent user. As the FDT protocol satisfies the simulation based definition, from the composition lemma [5], A semi-honest SP will have the similar behavior interacting with only an oracle which returns her half of the codeword. In this case, the SP wins only when she is able to break the innocent user security of Tardos code as shown in Theorem 2.1 in [26] that an innocent user will be framed with a probability no bigger than ϵ regardless of what biases are used and what is the pirate copy.

Lemma 2. *An innocent user will be accused with negligible probability by a semi-honest service provider assuming that the encryption* Enc_r *used is IND-CPA secure.*

Now we consider a malicious SP who may arbitrarily deviate from the protocol. With the simple checks, there is only one class of deviations left (which is not yet clear whether always detectable): the malicious SP submits different biases to the judge with those used during fing; This includes many subtle attacks, e.g., in one instance of FDT, the malicious SP uses the same messages in each pair to do the transmission, i.e., SP inputs $(m_0^{b_0}, m_0^{b_0}), (m_1^{b_1}, m_1^{b_1})$ (same for the hash values). Doing this the malicious SP will know the complete codeword of the user. Similarly, the SP could swap the messages in each pair, i.e., transmit version $1 - b_i$ if the code is b_i. Both of these behavior can be seen as special case of the above deviation. In the first case, the user codeword is essentially generated using a vector of probabilities $\langle p_1, \ldots, p_{2n} \rangle$, where each $p_i \in \{0, 1\}$, while the latter case is that each $p_i = 1 - p'_i$ where p'_i is the reported bias. As the judge will check the constancy of the codeword with the coin tossing, the more indices the SP reports different biases, the hight the probability she got caught. Through a careful analysis, we manage to show that the probability of accusing an innocent user and the probability of reporting different biases without being detected can never be non-negligible simultaneously, which further implies that either the malicious SP deviates without hurting the innocent user security, or the deviation will be almost always detected by the judge.

Theorem 1. *A malicious service provider can frame an innocent user without being detected by the judge with negligible probability if the underlying encryption* Enc_r *is IND-CPA secure, the commitment scheme is computationally binding, the digital signature scheme we use as black-box is existentially unforgeable, and the hash function is collision resistant.*

Security for the SP under group accusations. The analysis for the effectiveness of the accusation procedure will also proceed in two steps. We first deal with semi-honest users who will follow the fing protocol and the accusation procedure, but they will try to make a pirate copy and avoid being accused. From the half fingerprint known to the SP, the SP can always identify colluders. As the FDT satisfies the simulation based security, the behavior of the adversary is essentially the same as the one interacting with only an oracle returning the codewords of corrupted users, while no information about which half of the codewords are known to the SP is leaked. Further we can show that by relaxing the threshold on the judge side, whoever accused by the SP using the half fingerprint will also be accused by the judge using another half fingerprint.

Lemma 3. *Suppose* $\mathcal{U}_{cor}$*, with* $|\mathcal{U}_{cor}| \leq t$*, is a coalition of users. If all users are semi-honest during the* fing *protocol execution. The probability that no user is accused or an accused user is acquitted by the judge is* $\epsilon^{1/16} + \epsilon^{t/4} + \epsilon_0$*, where* ϵ *is the parameter from the Tardos code,* ϵ_0 *is negligible (to the security parameter), if the underlying commitment scheme* $\mathsf{Com}(\cdot)$ *is computationally hiding, and the encryption* Enc_s *is IND-CPA secure.*

The case that malicious users can arbitrarily deviate from the protocol are easier to analyze than Theorem 1 due to the simple checks. It is easy to see that in each round, the user is forced to be behave honestly, otherwise the deviation will be detected with overwhelming probability.

Theorem 2. *Suppose $\mathcal{U}_{cor}$, with $|\mathcal{U}_{cor}| \leq t$, is a coalition of users. Assuming Enc_s is IND-CPA secure, the commitment scheme $\mathsf{Com}(\cdot)$ is computationally hiding and the signature scheme used is existentially unforgeable, and the hash function is collision resistent. Then the probability that no user is accused or an accused user is acquitted by the judge is $\epsilon^{1/16} + \epsilon^{t/4} + \epsilon_0$, where ϵ is the error probability from the Tardos code, ϵ_0 is negligible (to the security parameter).*

5 Security Implications of Protocol Restarts

In the following, we consider the original Tardos code with length $m = 100t^2k$ and threshold $Z = 20tk$, where c is the number of colluders and $k = \log \frac{1}{\epsilon}$ the security parameter. For simplicity, we take t equal to the number of users n.

If the colluders are allowed restarts, they can act as follows. They do $(\mu - 1)$ restarts each to receive a total of μmn bits. For the pirate codeword, they output a zero whenever they can. Formally, for any $j \in [m]$, let x be the number of ones the pirates have received collectively at location j. They set y_j, the bit of the pirate copy at j, as follows.

$$y_j = \begin{cases} 1 & \text{if } x = \mu n; \\ 0 & otherwise. \end{cases}$$

We are going to show that with this simple strategy, each pirate escapes with high probability. To that end, let p denote the bias-vector, X denote the codeword of an arbitrary pirate, Y the pirate copy generated by the aforementioned strategy, and

$$U_j = \begin{cases} \sqrt{\frac{1-p_j}{p_j}}, & if X_j = 1; \\ -\sqrt{\frac{p_j}{1-p_j}}, & if X_j = 0. \end{cases}$$

The score of the pirate can be expressed as $S = \sum_{j \in [m]} Y_j U_j$. Our task is to upper-bound $\Pr[S > Z]$. We'll use $e^x \leq 1 + x + x^2$, valid for $x \leq 1$. Since $|U_j| < \sqrt{300n}$, choosing $\alpha < \frac{1}{10n}$ we have

$$\mathrm{E}[e^{\alpha S}] = \mathrm{E}[\prod e^{\alpha Y_j U_j}] = \prod \mathrm{E}[e^{\alpha Y_j U_j}] \leq \prod \mathrm{E}[1 + \alpha Y_j U_j + \alpha^2 Y_j^2 U_j^2]$$

$$\leq \prod (1 + \alpha \mathrm{E}[Y_j U_j] + \alpha^2 \mathrm{E}[U_j^2]) = \prod (1 + \alpha \mathrm{E}[Y_j U_j] + \alpha^2).$$

For any $j \in [m]$ we have

$$\mathrm{E}[Y_j U_j] = \mathrm{E}_{p_j}\left[p_j^{\mu n} \sqrt{\tfrac{1-p_j}{p_j}}\right] = \frac{1}{\pi/2 - 2t'} \int_{t'}^{\pi/2 - t'} \sin^{2\mu n - 1} r \cos r \; dr$$

$$= \frac{1}{\pi/2 - 2t'} \cdot \frac{1}{2\mu n} \cdot \sin^{2\mu n} r \Big|_{t'}^{\pi/2 - t'} = \frac{(1-t)^{\mu n} - t^{\mu n}}{(\pi - 4t')\mu n} \leq \frac{1}{3\mu n}.$$

Putting things together, and using $1 + x \leq e^x$, we obtain $\mathrm{E}[e^{\alpha S}] \leq \prod(1 + \alpha/(3\mu n) + \alpha^2) \leq e^{\alpha^2 m + \alpha m/(3\mu n)}$. An application of Markov's inequality now gives that:

$$\Pr[S > Z] < \frac{\mathrm{E}[e^{\alpha S}]}{e^{\alpha Z}} \leq e^{\alpha^2 m + \alpha m/(3\mu n) - \alpha Z}.$$

For $m = 100n^2k$, $Z = 20nk$, $k = \log(1/\epsilon)$, $\alpha = \frac{1}{10n}(1 - \frac{5}{3\mu})$, $\mu > 1$,

$$\Pr[S > Z] < \epsilon^{(1-\frac{5}{3\mu})^2}.$$

Thus, even allowing a single restart per user, is sufficient for the pirates to escape with high probability. Another way to view this, is that an instantiation of Tardos code that can handle a coalition of size t, is not secure against a coalition of size $2t$. The simple way around this, is to instantiate the code so as to handle coalition size μt, and allow each user at most $\mu - 1$ restarts.

6 Conclusion

In this paper, we constructed the first communication optimal asymmetric fingerprinting scheme, (i.e., the total number of bits transmitted in the protocol is almost the same as the length of the files), based on Tardos code. This is an appealing feature, especially for fingerprinting schemes in which large data (like movies) are transmitted. Besides rate optimality, we also considered two properties: security against accusation withdrawal and security under adversarial aborts, which are overlooked in previous asymmetric fingerprinting schemes.

Acknowledgements. The first, second and the fifth authors were supported by the ERC project CODAMODA. The third author was supported by the Estonian Research Council, and European Union through the European Regional Development Fund. The fourth author was supported by institutional research funding IUT20-57 of the Estonian Ministry of Education and Research.

References

1. Ambainis, A., Jakobsson, M., Lipmaa, H.: Cryptographic Randomized Response Techniques. In: Bao, F., Deng, R.H., Zhou, J. (eds.) PKC 2004. LNCS, vol. 2947, pp. 425–438. Springer, Heidelberg, Singapore (Mar 1–4, 2004) 3.1
2. Amiri, E., Tardos, G.: High Rate Fingerprinting Codes And the Fingerprinting Capacity. In: Mathieu, C. (ed.) SODA 2009. pp. 336–345. 3
3. Blake, I.F., Kolesnikov, V.: Strong Conditional Oblivious Transfer and Computing on Intervals. In: Lee, P.J. (ed.) ASIACRYPT 2004. Lecture Notes in Computer Science, vol. 3329, pp. 515–529. Springer (2004) 1.1, 2, 3.2
4. Boneh, D., Shaw, J.: Collusion-secure fingerprinting for digital data (extended abstract). In: Advances in Cryptology - CRYPTO '95, pp. 452–465 (1995) 1
5. Canetti, R.: Security and composition of multiparty cryptographic protocols. J. Cryptology 13(1), 143–202 (2000) 4.2

6. Charpentier, A., Fontaine, C., Furon, T., Cox, I.: An asymmetric fingerprinting scheme based on tardos codes. In: IH'11, pp. 43–58. (2011). 1, 1.1, 4, 4.1
7. Cleve, R.: Limits on the security of coin flips when half the processors are faulty (extended abstract). In: STOC. pp. 364–369. ACM (1986) 1.1
8. Damgård, I., Faust, S., Hazay, C.: Secure two-party computation with low communication. In: TCC. pp. 54–74 (2012) 1
9. Damgård, I., Jurik, M.: A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In:PKC. pp. 119–136. (2001) 2, 3.2, 4.1
10. Damgård, I., Zakarias, S.: Constant-overhead secure computation of boolean circuits using preprocessing. In: TCC. pp. 621–641 (2013) 1
11. Dodis, Y., Halevi, S., Rabin, T.: A Cryptographic Solution to a Game Theoretic Problem. In: Bellare, M. (ed.) CRYPTO 2000. pp. 112–130. 3.1
12. Fischlin, M.: A cost-effective pay-per-multiplication comparison method for millionaires. In: Topics in Cryptology - CT-RSA 2001, pp. 457–472 (2001) 2
13. Goldreich, O.: The Foundations of Cryptography - Volume 2, Basic Applications. Cambridge University Press (2004) 3.1
14. Ishai, Y., Kushilevitz, E., Meldgaard, S., Orlandi, C., Paskin-Cherniavsky, A.: On the power of correlated randomness in secure computation. In: TCC. pp. 600–620 (2013) 1
15. Ishai, Y., Paskin, A.: Evaluating Branching Programs on Encrypted Data. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 575–594. 2
16. Kiayias, A., Leonardos, N., Lipmaa, H., Pavlyk, K., Tang, Q.: Near Optimal Rate Homomorphic Encryption for Branching Programs. Tech. Rep. 2014 (2014), available at `http://eprint.iacr.org/2014/851` 1.1, 2, 3.2, 3.2
17. Kiayias, A., Pehlivanoglu, S.: Encryption for Digital Content, Advances in Information Security, vol. 52. Springer US (Oct 2010) 3
18. Laur, S., Lipmaa, H.: A new protocol for conditional disclosure of secrets and its applications. In: ACNS. pp. 207–225 (2007) 2
19. Lipmaa, H.: First CPIR Protocol with Data-Dependent Computation. In: ICISC 2009. pp. 193–210 (2009) 2
20. Naor, M., Nissim, K.: Communication preserving protocols for secure function evaluation. In: STOC. pp. 590–599 (2001) 1
21. Pfitzmann, B.: Trials of traced traitors. In: Anderson, R.J. (ed.) Information Hiding. pp. 49–64. Springer (1996) 1, 4
22. Pfitzmann, B., Schunter, M.: Asymmetric fingerprinting. In: Maurer, U.M. (ed.) EUROCRYPT. pp. 84–95. Springer (1996) 1, 1.1, 4, 4.1
23. Pfitzmann, B., Waidner, M.: Asymmetric fingerprinting for larger collusions. In: ACM Conference on Computer and Communications Security. pp. 151–160. ACM (1997) 1, 4
24. Rial, A., Balasch, J., Preneel, B.: A privacy-preserving buyer-seller watermarking protocol based on priced oblivious transfer. IEEE Transactions on Information Forensics and Security 6(1), 202–212 (2011) 1
25. Rial, A., Deng, M., Bianchi, T., Piva, A., Preneel, B.: A provably secure anonymous buyer-seller watermarking protocol. IEEE Transactions on Information Forensics and Security 5(4), 920–931 (2010) 1
26. Tardos, G.: Optimal probabilistic fingerprint codes. J. ACM 55(2) (2008) 1, 1.1, 3, 3.1, 4.1, 4.2