An Asymmetric Fingerprinting Code for Collusion-Resistant Buyer-Seller Watermarking

Serdar Pehlivanoglu
Computer Engineering
Zirve University
Gaziantep, Turkey
spehlivan38@gmail.com

ABSTRACT

A buyer seller watermarking protocol is intended to ensure copyright protection, privacy and security for both the buyer and seller simultaneously in an e-commerce application. A traitor should not be able to deny his responsibility of a copyright violation caused by him in such protocols. This feature is identified as non-repudiation in the literature. An efficient approach taken is through secure embedding of watermarks in an encrypted domain by using dither modulation and homomorphic encryption. To support non-repudiation along with collusion resistance, one needs an asymmetric collusion resistant fingerprinting code that is compatible with the watermarking technique. The design of such codes has not yet studied thoroughly with an exception of a recent work by Charpentier et. al. in [4].

In this paper, we propose an asymmetric binary fingerprinting code based on Boneh-Shaw code. When applied to the secure embedding of watermarks, we show that our code outperforms the code introduced by [4]: (i) we achieve constant communication round, (ii) we do not require any oblivious transfer protocol and (iii) we do not require any public Write Once Read Many (WORM) directory.

Categories and Subject Descriptors

K.4.4 [**Computers and Society**]: Electronic Commerce—*distributed commercial transactions; security*
; E.3 [**Data**]: Data Encryption

General Terms

Theory, Security

Keywords

Asymmetric Fingerprinting; Non-repudiation; Buyer-seller Watermarking

IH&MMSec'13, June 17–19, 2013, Montpellier, France.

1. INTRODUCTION

Fingerprinting Codes. A fingerprinting code is a mathematical tool to deter the unauthorized leakage of the protected content. An immediate application of a fingerprinting code is introduced by [5] and later formalized by [3] to perform key-distribution to differentiate the set of keys each receiver is assigned. Subsequent works (e.g. [22, 23, 24, 26]) on improving the code constructions have been very much related to the application of the key distribution and employed in the tracing mechanisms (including but not limited to [2, 5, 10, 11, 13, 15, 24, 25]).

An important consideration in the setting of fingerprinting systems is non-repudiation: the impossibility of a piracy collaborator(we call traitor) to deny his responsibility of a copyright violation caused by him. Indeed, it might be possible for a user that is accused to participate on a certain content leaking incident to deny her involvement and instead claim that is being framed by the tracing procedure. This scenario is plausible as in the context of such systems the accuser is the entity holding all system initialization parameters and as such is capable of fabricating a certain leakage incident that incriminates a user. This concern motivated the concept of asymmetric fingerprinting that was put forth in [18] and further elaborated on [1, 4, 19, 20].

Buyer Seller Watermarking Protocols. The enormous increase in the production of digital content is shaping our our everyday experiences, communication, entertainment, learning and commerce behaviors. The distribution of the digital content now requires a delicate handling as the content-consumers can more than ever engage in piracy and in re-distribution of the digital content. Watermarking has been employed for more than a decade to personalize the content so that the pirated copy reveals the identity of the users took role in the piracy (that we call traitor). The personalization would be possible by embedding buyers information (that uniquely binds the buyer) in the distributed content. In the classical setting, this requires a trust on the content seller (or in some cases producer or distributor) such that they would perform watermarking honestly or not distribute content illegally.

Buyer-seller watermarking protocols are introduced to weaken the trust assumption as a malicious seller may frame an innocent buyer or a buyer that is involved in a piracy may repudiate his guilty by invoking the possibility of framing by the seller. An essential sub-component of these protocols is the secure embedding of the watermarks. Introduced by Memon and Wong in [16], this is achieved by having secure watermarking in the encrypted domain for which an encryption scheme with additive homomorphism is employed.

Codes	*Coalition Size*	*Code Length*(ℓ)	*Buyer Enc.*	*Round Comp.*	*OT Cost*	*Additional Requirements*
[4]	w	$O(w^2 \log n/\varepsilon)$	$O(w^2 \log n/\varepsilon)$	$O(\ell)$	$O(\ell)$ many $OT_{1:N}$	Public WORM Directory of Size $N\ell$
[4]	n	$O(n^2 \log n/\varepsilon)$	$O(n^2 \log n/\varepsilon)$	$O(\ell)$	$O(\ell)$ many $OT_{1:N}$	Public WORM Directory of Size $N\ell$
Our Code (Section 2.4)	w	$O(w^4 \log n/\varepsilon)$	$O(w^2 \log n/\varepsilon)$	$O(1)$	None	None
Our Code (Section 2.2)	n	$O(n^3 \log n/\varepsilon)$	$O(n^2 \log n/\varepsilon)$	$O(1)$	None	None

Table 1: A comparison between the asymmetric fingerprinting codes: n is the number of codewords and N is the discretization factor of the probability distribution of Tardos code and ε is the security parameter.

There are different proposals for buyer-seller protocols: Many works (including but not limited to [9, 16]) rely on a trusted party that is called Watermarking authority which embeds the watermarking and sends to the buyer. Attempts to remove such trusted party leads to the proposals with double-watermarking techniques as in pretty much all earlier works including but not limited to [21, 27]. Double watermarking is discussed to be vulnerable to many deficiencies like quality degradation or ambiguity attacks. [6]. A more recent and efficient approach is based on dither modulation techniques and homomorphic cryptosystems as proposed in [14, 17] and further studied in the protocols of [7, 8].

Collusion-Resistant Buyer-Seller Watermarking It is believed that the secure watermarking protocols of [7, 8, 14, 17] can be extended to resist against a coalition of traitors with the application of a collusion resistant fingerprinting code. However, this puts forth a trust assumption as is stated[1] clearly in the work of [17] and this would not be immediately possible as is overlooked[2] in the work of [8].

The fingerprinting code that will be integrated into the secure watermarking technique should satisfy two more features: (i) being asymmetric, i.e. supporting the non-repudiation feature and (ii) being capable of cooperating with the dither modulation techniques of the above works. An asymmetric variant of Tardos code has recently published in [4] with a goal of satisfying these two features.

Our Results Our main result is to present an asymmetric binary fingerprinting code based on the Boneh-Shaw code. We provide for the first time a rigorous formalization of the asymmetric fingerprinting codes and describe our code in this model. We also present an efficient application of the code in the context of buyer seller watermarking protocols. Our code can be embedded in secure watermarking to strengthen the buyer-seller watermarking protocols with collusion resistance.

We make a comparison of our code with the asymmetric fingerprinting code of [4] in Table 1. The code of [4] requires heavy computation of $O(\ell)$ many $OT_{1:N}$ Oblivious Transfer protocols where N is the discretization factor of the probability distribution of Tardos code and ε is the security parameter. The number of rounds in the buyer-seller communication is also far away from being practical as it is linear in the length of the code. The code also requires a public Write Once Read Many (WORM) directory whose access is granted to all users. In comparison, the adaptation of our asymmetric fingerprinting code would not result in such efficiency losses.

[1]from [17]: "Although this (Authors' note: the trust assumption) might not be practical in real applications, it provides a theoretical solution to the problem of collusion."
[2]from [8]: "We focus on the watermarking embedding and detection scheme, but we will not explain anti-collusion fingerprints further."

2. AN ASYMMETRIC FINGERPRINTING BASED ON BONEH-SHAW CODE

In this section we present our asymmetric fingerprinting scheme that is based on the Boneh-Shaw code [3]. We first present the model in subsection 2.1, introduce our construction in 2.2, provide the security analysis in 2.3 and finally expand the length of the code with concatenation techniques in subsection 2.4.

2.1 The Model

A codeword x over an alphabet Q is an ℓ-tuple $\langle \mathsf{x}_1, \ldots, \mathsf{x}_\ell \rangle$ where $\mathsf{x}_i \in Q$ for $1 \leq i \leq \ell$. We call a set of codewords $\mathcal{C} \subseteq Q^\ell$ with size n by (ℓ, n, q)-code given that the size of the alphabet is q, i.e. $|Q| = q$.

Each codeword x in an (ℓ, n, q)-code $\mathcal{C}$ can be considered as providing a unique way of accessing to some specific object or functionality. In such setting, an adversary is modeled as corrupting a number of users (called *traitors*) and retrieving their codewords. The adversary, then, runs a **Forge** algorithm that produces a non-user codeword Q^ℓ that provides an access to the same functionality. This codeword is called *pirate codeword*. A fingerprinting code is designed to identify a traitor that took role in the forgery.

Fingerprinting codes are defined by two algorithms:

CodeGen(1^n) This algorithm outputs a pair $(\mathcal{C}, tk)$ where $\mathcal{C}$ is an (ℓ, n, q)-code with alphabet Q such that $|Q| = q$, and tk is a secret key to be used for identifying purposes.

Identify$(\mathcal{C}, tk, c)$ On input of a pirate codeword $c \in Q^\ell$, this algorithm either fails to identify, i.e. outputs $\perp$, or outputs a codeword index $t \in [n]$ which is accused of being an index of a traitor.

We say that the fingerprinting code is (α, w)-identifier, if the **Identify** algorithm, given a pirate codeword that is produced by a coalition of at most w traitors, is successful in identifying a traitor with probability at least $1 - \alpha$. We refer an interesting reader to the Chapter 1 of [12] for a comprehensive review of the fingerprinting codes.

Asymmetric Fingerprinting Code. An asymmetric fingerprinting code (introduced by [18]), in addition to the identifying capability, supports two additional features. Informally speaking, these features are (i) non-repudiation: a traitor can not deny its responsibility in the generation of a pirate codeword if it is indeed involved in such a piracy and (ii) non-framing: a malicious code designer can not frame an innocent user by distributing a pirate copy which incriminates that particular user.

In the asymmetric setting, the non-framing feature would be supported only if the code designer is not fully aware of any particular user codeword, since otherwise it would be easy to incriminate an innocent user. Producing a user codeword in cooperation with an additional private input from the user is one possible way of disallowing the code designer from the knowledge of the codeword. This cooperation can be implemented in various ways depending on the application, e.g. it can be considered as a secure two party computation and can be secured by generic methods like Yao's garbled circuit. We would like to isolate the conceptual definition of asymmetric fingerprinting from its implementation. Hence, we continue with formal definition and later introduce an asymmetric fingerprinting code that fits into our definitional framework. We finally discuss an application of our new code and its secure implementation in isolation from its formal definition.

An asymmetric fingerprinting code consists of three algorithms: code generation, identification and arbitration algorithms. We next define those algorithms:

AsymCodeGen$(1^n, \{X^i\}_{i\in[n]})$ The pair of $(\mathcal{C}, tk)$ is produced where $\mathcal{C}$ is an (ℓ, n, q)-code with alphabet Q such that $|Q| = q$, and tk is a secret key to be used for identifying purposes. The input X^i, for each $i \in [n]$, is used in the generation of the i-th codeword of the code $\mathcal{C}$.

A traitor, involving in the production of a pirate codeword, will inescapably put some footprints of its private input. The Identification algorithm, in addition to finding a traitor identity, discovers those footprints and computes some kind of a proof argument that is believed to reveal, otherwise impossible, information on the private input of the identified traitor. We call this proof by arbiter-proof as it will be the evidence of the proof of the traitor involvement in the case of arbitration. In this model, we define the Identification algorithm as follows:

AsymIdentify(tk, c) On input of a pirate codeword $c \in Q^\ell$, this algorithm either fails to identify and outputs $\perp$, or outputs a codeword index $t \in [n]$ along with an arbiter proof Ω. The t-th user will be accused of being a traitor.

The last algorithm **ArbiterPredicate** checks the validity of the pair (t, Ω) computed by the identification algorithm. We say the proof is valid if the arbiter proof Ω reveals some non-trivial information on the private input X^t of the t-th user. The arbitration procedure is based on some correlation computation between the user private information and the arbiter proof.

ArbiterPredicate$(tk, \{X^i\}_{i\in[n]}, t, \Omega)$ This predicate returns 1 if the proof Ω contains some non-trivial information on X^t and returns 0 otherwise.

Since X^i is private to the i-th user, it should be hard to produce a valid proof that results in a high correlation. This hardness implies that the tracer can not produce an arbiter proof for arbitrary user (i.e. non-framing feature) and a valid proof can not be refuted by an accused traitor (i.e. non-repudiation feature).

We are now ready to describe our asymmetric binary fingerprinting code based on Boneh-Shaw code. We provide a stand-alone description of our code, i.e. in an isolation from any possible application of the code.

2.2 The Construction

Asymmetric Code Generation: The input to the code generation algorithm is a collection of $\{X^i\}_{i\in[n]}$ which are supposed to be drawn independently and randomly from the set of all binary strings of length d where $d = 8n^2\lambda$ holds for some $\lambda = \ln(1/\varepsilon)$ that is the security parameter and n is the number of codewords. The algorithm will produce a code of length $\ell = d \cdot 2n$.

Similar to the Boneh-Shaw code, the **AsymCodeGen** algorithm first constructs a binary master-matrix $\mathbf{M}_d$ of size $n \times \ell$. In this construction, the i-th user input is embedded into the i-th row of the matrix as follows:

$$\mathbf{M}_d(i,j) = \begin{cases} 0, & if\ j < 2(i-1)d+1 \\ \mathrm{X}^i_{j-2(i-1)d} & if\ 2(i-1)d+1 \le j \le 2id \\ 1, & if\ j > 2id \end{cases}$$

where $\mathrm{X}^i = \langle \mathrm{X}^i_1, \mathrm{X}^i_2, \ldots, \mathrm{X}^i_d \rangle$ is a binary string of length d. The master-matrix is depicted in Figure 1, which consists of $2n$ blocks each of size d.

The code generation algorithm, then, samples a permutation $\pi \in_R \mathsf{Perm}(2dn)$ and permutes the columns of $\mathbf{M}_d$ according to the permutation π. The resulting matrix $\mathbf{M}_{d,\pi}$ would satisfy the following: $\mathbf{M}_{d,\pi}(i,j) = \mathbf{M}_d(i, \pi^{-1}(j))$. A codeword w^i for $1 \le i \le n$ is defined as an $2dn$-tuple where $\mathsf{w}^i_j = \mathbf{M}_{d,\pi}(i,j)$.

The output of the **AsymCodeGen** algorithm consists of π (which will serve as the tracing key) and the $(2dn, n, 2)$-code $\mathcal{C} = \{\mathsf{w}^1, \ldots, \mathsf{w}^n\}$.

Asymmetric Identification Algorithm: This algorithm has again some similarities with the original identification algorithm of the Boneh-Shaw code. Given a pirate codeword $\mathsf{p} \in \{0,1\}^{2dn}$ and π, it first applies the inverse permutation π^{-1} on p so that the resulting vector $\mathsf{x} \in \{0,1\}^{2dn}$ satisfies $\mathsf{x}_i = \mathsf{p}_{\pi(i)}$. The **AsymIdentify** algorithm will then partition x into $2n$ blocks of length d. We denote the i-th block by B_i and the number of 1's in B_i by k_i (we define $k_0 = 0$). As a notation we call k_i as the weight of block B_i. Before we present the algorithm formally, we first give some intuition.

Regarding the $2i, 2i+1$ and $(2i+2)$-th blocks we have the following observation: (i) these blocks consist of all 1's in rows from 1 to i, and (ii) these blocks consist all 0's in rows from $i+2$ to n. Consider a column that lies in one of these three blocks, the conclusion drawn from the observations is immediate: if $i+1$ is not in the traitor coalition, the permutation π will mask on which block the column lies in. Hence, regardless on how the traitor strategy is, we expect roughy equal weights in the pirate codeword blocks B_{2i}, B_{2i+1} and B_{2i+2}.

In addition to the above observation, if $k_2 > 0$ then the first user and if $k_{2n} < d$, then the n-th user can be accused of being a traitor. In other words, a wise traitor strategy (if the first or/and the n-th user is in the coalition) would try to have $k_2 = 0$ and $k_{2n} = d$. In this case, similar to

M	$[1,d]$	$[d+1,2d]$	$[2d+1,3d]$	$\ldots$	$[2id+1,(2i+1)d]$	$\ldots$	$[(2n-2)d+1,(2n-1)d]$	$[(2n-1)d+1,(2n)d]$
1	X^1	(1)	(1)	$\ldots$	(1)	$\ldots$	(1)	(1)
2	(0)	(0)	X^2	$\ldots$	(1)	$\ldots$	(1)	(1)
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$i+1$	(0)	(0)	(0)	$\ldots$	X^{i+1}	$\ldots$	(1)	(1)
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
n	(0)	(0)	(0)	$\ldots$	(0)	$\ldots$	X^n	(1)
Block	1	2	3	$\ldots$	$2i+1$	$\ldots$	$2n-1$	$2n$

Figure 1: The master matrix of the asymmetric binary fingerprinting code. (0) and (1) represents a binary string of all 0 and 1 respectively while X^i, for each $i \in [n]$, is a binary string of length d

the analysis discussed in Boneh-Shaw of [3], for the choice of $d = 8n^2\lambda$, eventually a non-negligible gap will occur between the weights of consecutive blocks B_{2i} and B_{2i+2} for some i value. This will let us to identify a traitor.

However, the challenge is still there, we need a proof that convinces the arbiter that identified user is indeed a traitor. First, recall that the $(2i+1)$-th block of the $(i+1)$-th user is the private input X^{i+1} of the user. If the weight k_{2i+1} of block B_{2i+1} is closer to k_{2i+2}, the difference between k_{2i} and k_{2i+1} will make us to believe that the 1's of the private input X^{i+1} is used extensively in the construction of the pirate block B_{2i+1}. In this case we set the arbiter proof Ω to be the pair $(B_{2i+1}, 1)$. On the contrary, if k_{2i+1} is closer to k_{2i}, the difference between k_{2i+1} and k_{2i+2} will make us to believe that the 0's of the private input X^{i+1} is used extensively in the construction of the pirate block B_{2i+1}. In this case we set the arbiter proof Ω to be the pair $(B_{2i+1}, 0)$.

Having said the intuition, we rigorously present the identification algorithm:

1. Find minimal $s \in \{0, 1, \ldots, n-1\}$ that satisfies $k_{2s+2} \geq 8n(s+1)\lambda$. Such choice will make the difference $k_{2s+2} - k_{2s} > 8n\lambda$. Hence, $s+1$ will be accused as a traitor index.
2. if $k_{2s+1} \geq (8ns + 2\sqrt{8ns} + 2)\lambda$ then set $\Omega = (B_{2s+1}, 1)$ as an arbiter proof for the accusation of the index $s+1$.
3. if $k_{2s+1} < (8ns + 2\sqrt{8ns} + 2)\lambda$ then set $\Omega = (B_{2s+1}, 0)$ as an arbiter proof for the accusation of the index $s+1$.
4. Otherwise fail.

We will formally prove, later in Section 4.1, that the above algorithm outputs a traitor index and its proof Ω is enough to convince our arbiter that we introduce next:

Arbiter Predicate In addition to the tracing key tk of the code and the collection of private inputs $\{X^i\}_{i\in[n]}$, the predicate takes an arbiter proof $\Omega = (arb, b) \in \{0,1\}^d \times \{0,1\}$ and the index $h \in \{1, \ldots, n\}$ of the accused user as input. The goal of the predicate is to check whether the proof Ω contains some non-trivial information on X^h or not.

We call a position is b-marked in a string if the string contains bit b in that position. For instance, the third position is a 1-marked in binary string 101000. Let us denote the number of b-marked positions in arb by p.

Assuming that X^h is chosen randomly and the string arb is chosen without any proof of tracing, we would expect to see an average $d/2$ number of matching positions of the strings arb and X^h. Similarly, we would expect to see an average of $p/2$ number of matchings in b-marked positions of the strings arb and X^h. Any substantial deviation from these expected numbers would be enough to be a proof of accusation since it implies that the string arb is not any string but reveals some non-trivial information on the private input X^h.

More specifically, the predicate checks the following conditions:

(i) computes the number of matchings in b-marked positions of the strings arb and X^h: say $\psi = |\{x \mid arb[x] = \mathrm{X}^h[x] = b\}|$. The predicate returns true if $|\psi - \frac{p}{2}| > \frac{\sqrt{\lambda p}}{16}$.

or

(ii) computes the number of matching positions of the strings arb and X^h: say $\omega = |\{x \mid arb[x] = \mathrm{X}^h[x]\}|$. The predicate returns true if $\omega \geq \frac{d}{2} + \frac{\sqrt{d\lambda}}{8}$.

We will formally prove, later in Section 4.1, that it is impossible to convince the arbiter by a guessing of the private input X^h.

2.3 Analysis of Our Code

As motivated and introduced in the model (Section 2.1), we require our code to satisfy traceability, non-repudiation and non-framing. Let us rephrase these features within our definitional framework:

Traceability: The **AsymIdentify** algorithm, outputting (t, Ω), is successful in identifying a traitor, i.e. t is among the traitor codeword indices.

Non-repudiation: The **ArbiterPredicate** returns true on input (t, Ω) with high probability, i.e. the proof Ω contains non-trivial information on the private input X^t.

Non-framing: A malicious tracer or code designer should not be able to produce an arbiter proof Ω' for any index t' that makes the arbiter predicate returns true for (t', Ω') even if it has access to all private inputs but $X^{t'}$, i.e. it should be difficult to compute an arbiter proof without actual evidence of piracy.

Our construction satisfies the following security claims regarding the above features:

THEOREM 1. *The binary asymmetric fingerprinting code presented in Section 2.2, for the choice of $d = 8\lambda n^2$, $\lambda = \ln 1/\varepsilon$ and $1 > \varepsilon > 0$, would be of length $\ell = 16n^3 \ln 1/\varepsilon$ and satisfies the following:*

- *Traceability: For any probabilistic polynomial time algorithm* **Forging** *and for any* $\mathsf{T} \subseteq [n]$*, it holds that*

$$\mathbf{Prob}[\{t\} \subseteq \mathsf{T}] \geq 1 - 2\varepsilon$$

where $(t, \Omega) \leftarrow \mathbf{AsymIdentify}(ik, \mathsf{p})$*, and* $\mathsf{p} \in \mathsf{desc}(\mathcal{C}_\mathsf{T})$ *is the output of the* **Forging** *algorithm that is given* $\{X^j\}_{j \in \mathsf{T}}$ *and* $\mathcal{C}_\mathsf{T} = \{\mathsf{c}^j \mid j \in \mathsf{T}\}$.

- *Non-repudiation: It further holds that*

$$\mathbf{Prob}[\mathbf{ArbiterPredicate}(\pi, \{(X^i)\}_{i\in[n]}, t, \Omega) = 1] \geq 1 - \varepsilon^{1/112}$$

- *Non-Framing: For any probabilistic polynomial time algorithm* **Framing** *and for any* $t' \in [n]$*, it holds that*

$$\mathbf{Prob}[\mathbf{ArbiterPredicate}(\pi, \{(X^i)\}_{i\in[n]}, t', \Omega') = 1] < 4\varepsilon^{1/384}$$

where Ω' *is an arbiter proof computed by the* **Framing** *algorithm on input* π *and* $\{(X^i)\}_{i\in[n]\setminus\{t'\}}$.

With the above parameters for any traitor coalition size, we say the code is $(2\varepsilon, \varepsilon^{1/112}, 4\varepsilon^{1/384}, n)$-identifer. If there is an upper bound w on the size of the traitor coalition $\mathsf{T} \subseteq [n]$, then we say the code is $(\cdot, \cdot, \cdot, w)$-identifier. For the readability of our paper, we leave the full proof of theorem 1 in Section 4.1. As a corollary, we present the above construction with simplifying parameters:

COROLLARY 1. *The binary asymmetric fingerprinting code presented in Section 2.2, for the choice of* $d = 8(384\lambda)n^2$*, ,* $\lambda = \ln 1/\varepsilon$ *and* $1 > \varepsilon > 0$*, is* $(\varepsilon, \varepsilon, \varepsilon, n)$*-identifer in the sense of Theorem 1. The length of the code is* $\ell = 6144n^3 \ln 1/\varepsilon$*. We say simply it is* (ε, n)*-identifier.*

The proof is straightforward given that $2\varepsilon^{384} < 2\varepsilon^3 < 4\varepsilon$, which is correct for any reasonable choice of ε, i.e. $\varepsilon < 1/2$. However, note that this simplifying parameters increases the constant factor in the length of the code and presented in here for simply illustrative purposes.

2.4 Code Concatenation

As in the case of original Boneh-Shaw code [3], the length of our asymmetric fingerprinting code is cubic in the number of codewords. Generating a code with shorter length and smaller alphabet size is a common challenge in the design of fingerprinting codes. Code concatenation is a technique utilized extensively in coding theory and is also proven effective in shortening code lengths in the domain of fingerprinting codes.

Code concatenation entails the composition of two codes: an 'inner' code with an 'outer' code. The composition is feasible as long as the codes adhere to a suitable structural characteristic. The end effect is that the codewords of the inner code substitute the alphabet symbols of the outer code. In general, the inner code is chosen to be resistant against any size of traitor coalition, while the the resistance of outer code is parameterized by some w value. The concatenated code supports collusion resistance against traitor coalitions of size less than or equal to w.

We concatenate our asymmetric fingerprinting code (as an inner code) with a w-identifier Chor-Fiat-Naor fingerprinting code[5] (as an outer code), we obtain a shorter asymmetric fingerprinting code. More precisely, we obtain:

THEOREM 2. *There is a concatenated asymmetric binary fingerprinting code of length* $O(w^4 \log \frac{w}{\varepsilon} \log \frac{n}{\varepsilon})$ *which is* $(\varepsilon, w)-$*identifier (i.e.* $(\varepsilon, \varepsilon, \varepsilon, w)$*-identifier in the sense of Theorem 1). We use our code presented in Corollary 1 as inner code, and secret Chor-Fiat-Naor code from [5] as outer code.*

We elaborate on the details of the concatenation and present the proof of Theorem 2 in Section 4.2.

3. APPLICATION: BUYER-SELLER WATERMARKING PROTOCOL

3.1 Secure Watermarking in the Encrypted Domain

In this section, we provide the basis of the secure watermarking in the encrypted domain which is based on dither modulation techniques and homomorphic cryptosystems as proposed in [14, 17] and employed in the protocols of [7, 8].

We assume that a vector of host features x, of size m, has been extracted from the original content and we denote the i-th feature by x_i for $i \in [m]$. Out of these host features some number of them, say ℓ of them, are chosen by the content owner: denoting the chosen set of indices by $C = \{c_1, \ldots, c_\ell\} \subseteq [m]$, the corresponding watermarked features, for $i \in [\ell]$, using a scalar binary dither modulation can be expressed as

$$y_{c_i} = f(x_{c_i}, x) + w_i \cdot \Delta(x_{c_i}, x)$$

where $f(x_{c_i}, x)$ and $\Delta(x_{c_i}, x)$ denoting respectively a suitable function of the original feature and a signal dependent quantization step, change according to the chosen embedding technique. Here $w_i \in \{0, 1\}$ is the bit embedded in the content. Not to interfere with our main results, we refer reader to [14, 17] for the actual choices of f and Δ functions.

The watermarked features above are not suitable for processing through a homomorphic system since they are represented as real values. An integer valued watermarked feature is the obtained as

$$\begin{aligned} z_{c_i} &= \lceil f(x_{c_i}, x) \cdot Q \rfloor + w_i \cdot \lceil \Delta(x_{c_i}, x) \cdot Q \rfloor \\ &= f_Q(x_{c_i}, x) + w_i \cdot \Delta_Q(x_{c_i}, x) \end{aligned}$$

where $\lceil \cdot \rfloor$ is the rounding function and Q is a scale factor that can be adjusted according to the required precision. By assuming an additively homomorphic cryptosystem, the above equation can be translated into the encrypted domain as follows where $E(\cdot)$ is an additively homomorphic public key cryptosystem:

$$E(z_{c_i}) = E(f_Q(x_{c_i}, x)) \cdot E(w_i)^{\Delta_Q(x_{c_i}, x)}$$

In a typical application of the above technique, we assume that the buyer possesses the secret key of the homomorphic encryption scheme. The buyer watermark $E(w_i)$ is produced by the buyer and transmitted to the seller. The seller, being the content owner, knows the plaintext version of x and can compute both $f_Q(x_{c_i}, x)$ and $\Delta_Q(x_{c_i}, x)$ in clear and compute the above equation relying only on the homomorphic properties of the underlying cryptosystem. Finally, the buyer can decrypt and retrieve z_{c_i} that is the watermarked host feature. The protocol of [8] improves this basic idea with a joint watermarking of the buyer and seller to support the ground for non-repudiation and non-framing features.

Remarks: As discussed above, only ℓ features out of m host features are chosen to be watermarked. The chosen features should be kept secret to ensure the robustness of the watermarking. For this purpose, all host features are transmitted in the encrypted form. To decrease the computation cost, a practical implementation with a composite signal representation is given in [7]. For simplification of the further presentation, we consider as if all host features are used in the watermarking process without loss of generality.

3.2 Collusion-Secure Watermarking

In the early works of [7, 8, 14, 17], the watermarking schemes are not collusion resistant; i.e. a number of corrupted buyers may be able to remove the watermark by comparing or composing their differently watermarked copies or they may produce a different copy not initially available to them. This is overlooked in those works by simply referring to anti-collusion fingerprinting or collusion secure codes like [3] or [26].

However, an immediate application of these conventional collusion resistant fingerprinting codes do not support non-repudiation and will not be very much useful in the buyer-seller watermarking protocol. An asymmetric variant of Tardos code has recently published in [4] with a goal of fulfilling the need on collusion resistant asymmetric fingerprinting code. Our novel binary asymmetric fingerprinting code of Section 2 can be efficiently employed in the context of buyer-seller watermarking to support collusion resistance and non-repudiation. In the next section we detail the application and compare it with the work of [4].

3.3 Secure Watermarking with Our Asymmetric Fingerprinting Code

In this section, we apply our collusion resistant asymmetric fingerprinting in the context of secure watermarking embedding. For the simplicity, we consider the basic fully-collusion resistant code. We will follow a joint watermarking of our code into the host features of the content as follows:

Setup: As an initialization, the seller picks $\ell = 2n \times d$ number of host features from the content. The same set of host features will be used for every buyer of the content. The seller also specifies a permutation π. Each buyer will randomly choose his/her private input: for the $i-th$ user we denote it by a binary string $X^i = X_1^i X_2^i \dots X_d^i$ of length $d = 8n^2\lambda$. Each user also picks a pair of public-secret keys for an additively homomorphic public key cryptosystem $E(\cdot)$.

After the above setup, we execute a 2-round secure watermarking that takes place between a buyer, say the i-th buyer, and the seller:

Buyer Watermark: The i-th buyer will compute the encryption of each bit of its private input, i.e. computes $E(X_k^i)$ for $k = 1, \dots, d$ with its public key. These bits are transmitted to the seller.

Seller Watermark: Adapting the master matrix (see figure 1) of our asymmetric fingerprinting code, the seller first computes the following watermarks for the i-th buyer:

$$E(w_j) = \begin{cases} E(0), & if\ j < 2(i-1)d+1 \\ E(\mathrm{X}^i_{j-2(i-1)d}), & if\ 2(i-1)d+1 \le j \le 2id \\ E(1), & if\ j > 2id \end{cases}$$

The seller then permutes the encrypted watermarks and embeds the j-th watermark into the $\pi(j)$-th host-feature. This step matches with the permutation step our code generation algorithm. At the end of this step, the seller would have

$$E(z_j) = E(f_Q(x_j, x)) \cdot E(w_{\pi^{-1}(j)})^{\Delta_Q(x_j, x)}$$

for the j-th host feature.

The seller transmits the watermarked content to the buyer. The buyer, by using his secret key, will be able to decrypt $E(z_j)$, for $j = 1, \dots \ell$, and obtains his watermarked copy. Upon observation of a pirate copy of the content, the seller extracts the watermarks from the host features of the pirate copy. These made up a pirate codeword $\mathsf{p} \in \{0,1\}^\ell$. We call our asymmetric identification algorithm on input p and identify a traitor along with an arbiter proof.

As it is immediate in the above procedure, the watermarking protocol above satisfies both non-repudiation and non-framing by inheriting those features from our asymmetric code. The adaptation of our code requires no extra effort, simply fits into the model of secure watermark embedding of [7, 8, 14, 17]. The protocol consists of a constant round of communication, and $O(n^2)$ many encryptions on both of the sides. The concatenated code of Section 2.4 can also be adapted to the above secure watermarking protocol: this will provide a trade-of between the size of the traitor coalition and the code length (and the number of encryptions processed on the buyer side).

We make a comparison of our code with the asymmetric fingerprinting code of [4]. (see Table 1 in the Introduction) The code of [4] requires heavy computation of $O(\ell)$ many $OT_{1:N}$ Oblivious Transfer protocols where N is the discretization factor of the probability distribution of Tardos code. The number of rounds in the buyer-seller communication is also far away from being practical as it is linear in the length of the code. The code also requires a public Write Once Read Many (WORM) directory whose access is granted to all users.

In comparison, the adaptation of our asymmetric fingerprinting code would not result in such efficiency losses. The only drawback of our system, as is the case in the variants of the Boneh-Shaw codes, is that the code is longer compared to the Tardos based asymmetric fingerprinting code of [4]. Such increase in the length can be tolerable for sufficiently large host data. Despite the longer length, we do not require the buyer to prepare encryptions linear in the length of the code: fortunately, the number of encryptions at the buyer side (i.e. that is the length of the private input X) is same with the number of buyer-encryptions in the code of [4].

Remark on Bit Encryption: One crucial remark is on the reliability of the Buyer sending encryption of the bits. This kind of issue appears in all Buyer-Seller protocols: it can be addressed by the use of zero-knowledge protocols as in the example of [7, 8]. We can avoid this complexity by simply forcing the Buyer to reveal a portion of its encrypted-bits so that the Buyer is caught in the case of cheating. Asking for half of the bits to be revealed will increase the size of the private-input by a factor of two while failure of catching a cheating Buyer would be bounded by a probability of 2^{-k} for k many non-bit encryptions. The rest of the non-revealed portion of the private input can be used in joint computation of the watermark embedding procedure.

4. PROOFS

4.1 The Proof of Theorem 1

We next prove the above theorem separately for each property specified above. In the proof, we will frequently utilize the exponentially decreasing bounds on the tails of a class of related distributions commonly referred to as Chernoff Bounds. Whenever we need these tails, we will refer to them

in the text and leave the detailed discussions to e.g., Chapter 1 of [12]. Before proceeding we present a general lemma that will be useful throughout the proof.

LEMMA 1. *We sample a binary string of length $2d$ from the set of strings with Hamming weight k. Let k_L and k_R be the hamming weight of left and right halves respectively. $k_R - k_L \geq 2\sqrt{\frac{k}{2}\lambda}$ holds with probability at most $2e^{-\lambda}$.*

PROOF. Suppose that we have $k_R - k_L \geq 2\sqrt{\frac{k}{2}\lambda}$, then it holds that $k_L \leq \frac{k}{2} - \sqrt{\frac{k}{2}\lambda}$. We next prove that the latter happens with probability at most $e^{-\lambda}$.

Denote the random variable X that is the weight k_L conditioning on the event that the total Hamming weight is k. The probability that $X = r$ equals:

$$\mathbf{Prob}[X = r] = \frac{\binom{d}{r}\binom{d}{k-r}}{\binom{2d}{k}}$$

Consider a random variable Y which is a binomial distribution of k successive experiments with success probability $1/2$, hence $\mathbf{Prob}[Y = r] = \frac{\binom{k}{r}}{2^k}$. It is easy to see the following(just substitute and do regular computation):

$$\mathbf{Prob}[X = r] \leq 2 \cdot \mathbf{Prob}[Y = r]$$

Note that $E[Y] = k/2$. By applying the Chernoff bound we have the following for some $0 < \alpha < k/2$.

$$\mathbf{Prob}[Y \leq k/2 - \alpha] \leq e^{-2\alpha^2/k}$$

Substituting $\alpha = \sqrt{\frac{k\lambda}{2}}$, we will see with what probability the left half has a weight $\leq k/2 - \sqrt{\frac{k\lambda}{2}}$:

$$\begin{aligned}\mathbf{Prob}[X \leq k/2 - \sqrt{\tfrac{k\lambda}{2}}] &\leq 2 \cdot \mathbf{Prob}[Y \leq k/2 - \sqrt{\tfrac{k\lambda}{2}}] \\ &\leq 2 \cdot e^{\frac{-2}{k} \cdot \frac{k\lambda}{2}} \\ &= 2 \cdot e^{-\lambda}\end{aligned}$$

This will complete the proof. ■

Traceability: Correctness of the Identification. We, first, argue that the **AsymIdentify** algorithm outputs an index from the set $[n]$.

LEMMA 2. *The* **AsymIdentify** *algorithm always outputs an index from the set $[n]$.*

PROOF. The proof is straightforward given the fact that $d = 8\lambda n^2$. The index n will be output in the worst case. ■

Let s be the minimal index computed by the **AsymIdentify** algorithm that satisfies $k_{2s+2} \geq 8n(s+1)\lambda$, it further holds that $k_{2s} < 8ns\lambda$ (to count the case $s = 0$ we say $k_{2s} \leq 8ns\lambda$ without loss of generality). We have two cases either (i) $k_{2s+1} \geq (8ns + 2\sqrt{8ns} + 2)\lambda$ or (ii) $k_{2s+1} < (8ns + 2\sqrt{8ns} + 2)\lambda$. In both cases the user $s+1$ is accused. We will study both of these cases separately and argue the failure probability in accusation.

(i) Case $k_{2s+1} \geq (8ns + 2\sqrt{8ns} + 2)\lambda$ holds and user $s+1$ is accused. Observe from the master matrix in Figure 1 that if the user-codeword w^{s+1} is not available to the traitor coalition, then the traitors will not be able to differentiate the block B_{2s+1} and B_{2s}: in both of these blocks traitors with smaller indices are marked with 1 while the traitors with larger instances are marked with 0. Hence, $|k_{2s+1} - k_{2s}|$ is expected to be close to 0. Fortunately we have $k_{2s+1} \geq (8ns + 2\sqrt{8ns} + 2)\lambda$ and $k_{2s} \leq 8ns\lambda$.

Let $k_{2s+1} + k_{2s} = k$. We next claim that the difference $k_{2s+1} - k_{2s}$ is at least $2\sqrt{\frac{k}{2}\lambda}$:

Let $k_{2s} = a^2\lambda$ holds for some $a < \sqrt{8ns}$. The claim is easily satisfied for the $s = 0$ or $a = 0$ cases: indeed we have $k_{2s} = 0$ and $k_{2s+1} = k \geq 2\lambda$ for which $k \geq 2\sqrt{\frac{k}{2}\lambda}$ holds trivially. Hence we investigate the $s > 0$ and $a > 0$ cases.

If $a > 0$ holds, then we have $k_{2s+1} = (a^2 + 2a + y)\lambda$ for some $y \geq 2$ which is obvious based on the fact that $k_{2s+1} \geq (8ns + 2\sqrt{8ns} + 2)\lambda$. We then have $\frac{k}{2} = (a^2 + a + y/2)\lambda$.

$$\begin{aligned}k_{2s+1} - k_{2s} &= (2a + y)\lambda \\ &= \sqrt{(4a^2 + 4ay + y^2)\lambda^2} \\ &\geq \sqrt{(4a^2 + 4a + 2y)\lambda^2} \\ &\geq 2\sqrt{(a^2 + a + y/2)\lambda^2} \\ &\geq 2\sqrt{\tfrac{k}{2}\lambda}\end{aligned}$$

The above can happen with probability at most $2e^{-\lambda} = 2\varepsilon$ as it is implied by Lemma 1. The conclusion is immediate: the codeword w^{s+1} is a traitor codeword with probability at least $1 - 2\varepsilon$. The arbiter proof Ω is set to be $(B_{2s+1}, 1)$. Note for future reference that B_{2s+1} would have k_{2s+1} many 1-marked positions.

(ii) Case $k_{2s+1} < (8ns + 2\sqrt{8ns} + 2)\lambda$ holds and user $s+1$ is accused. In this case, we consider the number of 0-marked positions in the blocks B_{2s+1} and B_{2s+2}. Denoting the numbers by m_{2s+1} and m_{2s+2}, they satisfy $m_{2s+1} = 8n^2\lambda - k_{2s+1}$ and $m_{2s+2} = 8n^2\lambda - k_{2s+2}$. Considering the assumed constraints on k_{2s+1} and k_{2s+2} we obtain $m_{2s+1} \geq (8n^2 - 8ns - 2\sqrt{8ns} - 2)\lambda$ and $m_{2s+2} \leq (8n^2 - 8n(s+1))\lambda$.

As before if the user-codeword w^{s+1} is not available to the traitor coalition, then the traitors will not be able to differentiate the block B_{2s+2} and B_{2s+1}: in both of these blocks traitors with smaller indices are marked with 1 while the traitors with larger instances are marked with 0. Hence, $|m_{2s+2} - m_{2s+1}|$ is expected to be close to 0 unless w^{s+1} is a traitor codeword.

Let $m_{2s+2} + m_{2s+1} = m$. We next claim that the difference $m_{2s+1} - m_{2s+2}$ is at least $2\sqrt{\frac{m}{2}\lambda}$:

Let $m_{2s+2} = a^2\lambda$ holds for some $a < \sqrt{8n^2 - 8n(s+1)}$. The claim again is easily satisfied for the case $a = m_{2s+2} = 0$ case: indeed we have $m_{2s+1} = m \geq (8n^2 - 8n(n-1) - 2\sqrt{8n(n-1)} - 2)\lambda = (8n - 2\sqrt{8n^2 - 8n} - 2)\lambda$. A further analysis will lead to the fact that $m \geq 2\lambda$ for which $m \geq 2\sqrt{\frac{m}{2}\lambda}$ holds trivially. Hence we investigate the $a, m_{2s+2} > 0$ cases.

If $a > 0$ holds, then we next prove that $m_{2s+1} = (a^2 + 2a + y)\lambda$ for some $y \geq 2$:

$$\begin{aligned}m_{2s+1} - m_{2s+2} &\geq (8n - 2\sqrt{8ns} - 2)\lambda \\ (2a + y)\lambda &\geq (8n - 2\sqrt{8ns} - 2)\lambda \\ 2\sqrt{8n^2 - 8n(s+1)} + y &\geq 8n - 2\sqrt{8ns} - 2 \\ y &\geq 8n - 2 - 2(\sqrt{8ns} + \sqrt{8n^2 - 8n(s+1)}) \\ y &\geq 8n - 2 - 2\sqrt{2(8ns + 8n^2 - 8n(s+1))} \\ y &\geq 8n - 2 - 2\sqrt{16n^2 - 16n} \\ y &\geq 8n - 2 - 2\sqrt{16n^2 - 16n + 4} \\ y &\geq 8n - 2 - 2(4n - 2) \\ y &\geq 2\end{aligned}$$

Based on the above computation we obtain $\frac{m}{2} = (a^2+a+y/2)\lambda$ for $y \geq 2$. Hence, the difference $m_{2s+1} - m_{2s+2}$ is at least $2\sqrt{\frac{m}{2}\lambda}$ based on exact same analysis made in the first case. This can happen with probability with at most $2e^{-\lambda} = 2\varepsilon$ as it is implied by Lemma 1(a variant of the lemma). The conclusion is immediate: the codeword w^{s+1} is a traitor codeword with probability at least $1-2\varepsilon$. The arbiter proof Ω is set to be $(B_{2s+1}, 0)$. Note for future reference that B_{2s+1} would have m_{2s+1} many 0-marked positions.

Non-repudiation: Convincing the Arbiter. We consider the arbiter proof $\Omega = (B_{2s+1}, \sigma)$ produced above for the traitor index $s+1$. Based on the analysis above, the number of σ-marked positions in block B_{2s+1} is equal to $Q \stackrel{def}{=} (a^2+2a+y)\lambda$ for some $0 < a < \sqrt{8n^2}$ and $y \geq 2$ while the number of σ-marked positions in block $B_{2s+2\tau}$ is equal to $L \stackrel{def}{=} a^2\lambda$ where $\tau = 1-\sigma$. For simplicity of notation, we also define $S \stackrel{def}{=} 8n^2\lambda$.

To convince the arbiter that $s+1$ is a traitor index (we denote the accused traitor by T), we need to prove one of the following: either (1) the number of matches in σ-marks between B_{2s+1} and X^{s+1} is outside the interval $[\frac{Q}{2} - \frac{\sqrt{\lambda Q}}{16}, \frac{Q}{2} + \frac{\sqrt{\lambda Q}}{16}]$ or (2) the number of matches in any mark(0 or 1 marks) between B_{2s+1} and X^{s+1}, i.e. the hamming weight of $B_{2s+1} \oplus X^{s+1}$, is at least $\frac{S}{2} + \frac{\sqrt{S\lambda}}{8}$.

Before we continue let us fix some notations to make the analysis easier. We denote $p_{i,j}$ by the cardinal/size of the set $\{x \in [d] \mid B_{2s+1}[x] = i \wedge X^{s+1}[x] = j\}$ for $(i,j) \in \{0,1\}^2$. $p_{i,j}$ is essentially the number of positions where B_{2s+1} has i-mark while X^{s+1} has j-mark.

The requirements for convincing the arbiter(non-repudiation) in terms of the new notation can be restated as follows: either (1) $p_{\sigma,\sigma}$ is outside the interval $[\frac{Q}{2} - \frac{\sqrt{\lambda Q}}{16}, \frac{Q}{2} + \frac{\sqrt{\lambda Q}}{16}]$ or (2) $p_{\sigma,\sigma} + p_{\tau,\tau}$ is at least $\frac{S}{2} + \frac{\sqrt{S\lambda}}{8}$.

We proceed with proof by contradiction. Let us assume that arbiter proof $\Omega = (B_{2s+1}, \sigma)$ is not enough to convince the arbiter. Hence, we assume that:

$$\frac{Q}{2} + \frac{\sqrt{\lambda Q}}{16} \geq p_{\sigma,\sigma} \geq \frac{Q}{2} - \frac{\sqrt{\lambda Q}}{16}$$
$$p_{\sigma,\sigma} + p_{\tau,\tau} < \frac{S}{2} + \frac{\sqrt{S\lambda}}{8}$$

Since the number of σ-marked positions in block B_{2s+1} is equal to Q, we obtain $p_{\sigma,\tau} + p_{\sigma,\sigma} = Q$. This will result in the following:

$$\frac{Q}{2} + \frac{\sqrt{\lambda Q}}{16} \geq p_{\sigma,\tau} \geq \frac{Q}{2} - \frac{\sqrt{\lambda Q}}{16}$$
$$p_{\sigma,\tau} - p_{\sigma,\sigma} < \frac{\sqrt{\lambda Q}}{8} \leq \frac{\sqrt{S\lambda}}{8}$$

From above, we obtain $Q_\tau \stackrel{def}{=} p_{\sigma,\tau} + p_{\tau,\tau} < \frac{S}{2} + \frac{\sqrt{S\lambda}}{4}$. We next argue that this can happen only with probability ..... .

At this phase of the proof, we introduce another notation for brevity: we denote the j-th block of an i-th receiver in the master matrix of the figure 1 by $M_{i,j}$ for $i \in [n]$ and $j \in [2n]$. The master matrix satisfies the following property:

If $j \leq 2i-2$ then $M_{i,j}$ consist of all 0-marks.
If $j \geq 2i$ then $M_{i,j}$ consists of all 1-marks
$M_{i,2i-1}$ is equal to the private input X^i.

Due to the property above, the number of τ-marked positions in block $M_{s+1,2s+2\tau}$ is equal to S and the number of τ-marked positions in block $M_{s+1,2s+1} = X^{s+1}$ is equal to $Q_\tau = p_{\sigma,\tau} + p_{\tau,\tau}$.

Considering again the property of the master matrix presented above, let us elaborate on the view of traitors on the $(2s+1)$-th and $(2s+2\tau)$-th blocks of the master matrix. Any traitor with a smaller index $k < s+1$ are assigned full σ-marks in blocks $M_{k,2s+2\tau}$ and $M_{k,2s+1}$, while any traitor with a larger index $l > s+1$ are assigned full τ-marks in both of the blocks $M_{l,2s+2\tau}$ and $M_{l,2s+1}$.

Since, the columns of the master matrix is permuted randomly during the code generation, the τ-marked positions in the blocks of $M_{s+1,2s+2\tau}$ and $M_{s+1,2s+1}$ are masked with that permutation. In other words, the traitor coalition will not be able to differentiate a τ-marked position in block $M_{s+1,2s+2\tau}$ from a τ-marked position in block $M_{s+1,2s+1}$.

Let us call a position/column in $(2s+1)$-th or $(2s+2\tau)$-th block to be good, if that position is σ-marked in pirate copy (i.e. $B_{2s+2\tau}$ or B_{2s+1}, say B-blocks) and τ-marked in the original blocks of the $s+1$-th user (i.e. $M_{2s+2\tau}$ or M_{2s+1}, say M-blocks). The distribution of the good positions over the pirate B-blocks is expected to be equal to the distribution of the τ-marked positions over the user M-blocks. This is true because of the indistinguishability of τ-marks in M-blocks discussed above.

Following our notation, the number of good positions in block $B_{2s+2\tau}$ is equal to L and the number of good positions in block B_{2s+1} is equal to $p_{\sigma,\tau}$. From earlier discussions, we know that $p_{\sigma,\tau} > \frac{Q}{2} - \frac{\sqrt{\lambda Q}}{16} > \frac{(a^2+a)\lambda}{2} = \frac{L+\sqrt{L\lambda}}{2}$.

Let us fix the total number of good positions to be K, i.e. $K \stackrel{def}{=} L + p_{\sigma,\tau} > \frac{3L+\sqrt{L\lambda}}{2}$. K good positions are chosen by the traitor coalition from a total of $S + Q_\tau$ many τ-marked positions of M-blocks. Let us denote the probability of observing a good position in block $B_{2s+2\tau}$ by q, we have $q = \frac{S}{S+Q_\tau}$. Applying what we already know, we obtain:

$$q = \frac{S}{S+Q_\tau} > \frac{4S}{6S+\sqrt{S\lambda}} > \frac{4L}{6L+\sqrt{L\lambda}}$$

Let Z be a random variable that is the number of good positions in block $B_{2s+2\tau}$: we define $Prob[Z \leq L | K, q]$ to be the probability of observing at most L good positions conditioned on the probability q and the total number of good positions K. If we consider $\frac{3L+\sqrt{L\lambda}}{2} < K$ many trials, we would have:

$$Prob[Z \leq L | K, q] \leq Prob\left[Z \leq L | \frac{3L+\sqrt{L\lambda}}{2}, q\right]$$

If we further decrease the probability q to $\frac{4L}{6L+\sqrt{L\lambda}}$, we obtain:

$$Prob[Z \leq L | K, q] \leq Prob\left[Z \leq L | \frac{3L+\sqrt{L\lambda}}{2}, \frac{4L}{6L+\sqrt{L\lambda}}\right]$$

We next apply the Chernoff bound that holds

$$\mathbf{Prob}[Z \leq \mu - F] < e^{-\frac{F^2}{2\mu}}$$

where μ is the expected number and F is the deviation from the expected number. Setting $L = \mu - F$ and $\mu = \frac{3L+\sqrt{L\lambda}}{2} \cdot \frac{4L}{6L+\sqrt{L\lambda}}$ we compute $F = \frac{L\sqrt{L\lambda}}{6L+\sqrt{L\lambda}}$. We next bound the probability, by bounding $\frac{F^2}{\mu}$:

$$\begin{aligned}
\frac{F^2}{\mu} &= \frac{L^2 L\lambda}{6L+\sqrt{L\lambda}} \cdot \frac{2(6L+\sqrt{L\lambda})}{4L(3L+\sqrt{L\lambda})} \\
&= \frac{1}{2} \cdot \frac{L^2\lambda}{(6L+\sqrt{L\lambda})(3L+\sqrt{L\lambda})} \\
&= \frac{1}{2} \cdot \frac{a^4\lambda^3}{(6a^2\lambda+\sqrt{a^2\lambda^2})(3a^2\lambda+\sqrt{a^2\lambda^2})} \\
&= \frac{\lambda}{2} \cdot \frac{1}{(6+1/a)(3+1/a)} \\
&\geq \frac{\lambda}{56}
\end{aligned}$$

Hence, we bound the error probability by $e^{-\frac{\lambda}{112}} = \epsilon^{1/112}$

Non-framing: Hardness of Framing an Innocent. The **Framing** algorithm produces an arbiter proof $(arb_{t'}, b) \in \{0,1\}^d \times \{0,1\}$ for index $t' \in \{1,\ldots,n\}$ independently from the binary string $\mathrm{X}^{t'}$. Let the number of positions with b in arb be p. The predicate then computes the number of matches with the string $arb_{t'}$ and the input string $\mathrm{X}^{t'}$. The predicate returns true if and only if one of the following holds:

(i) If the number of the matches over b-positions substantially deviate from the expected average of $p/2$, i.e. if either $|\{x \mid arb[x] = \mathrm{X}^{t'}[x] = b\}| < \frac{p}{2} - \frac{\sqrt{p\lambda}}{16}$ or $|\{x \mid arb[x] = \mathrm{X}^{t'}[x] = b\}| > \frac{p}{2} + \frac{\sqrt{p\lambda}}{16}$ holds.

or

(ii) If the total number of matches substantially exceeds $d/2$, i.e. $|\{x \mid arb[x] = \mathrm{X}^{t'}[x]\}| \geq \frac{d}{2} + \frac{\sqrt{d\lambda}}{8}$.

We will bound the probability of the above conditions hold by using the Chernoff bounds applying on the positions we make a comparison.

We list the positions where we check for the matching between $arb_{t'}$ and $\mathrm{X}^{t'}$: denoting the sequence of positions by $j_1,\ldots,j_s$, we define Y_{j_i} to be a random variable such that $\mathrm{Y}_{j_i} = 1$ if $arb_{t'}[j_i] = \mathrm{X}_{t'}[j_i]$ and $\mathrm{Y}_i = 0$ otherwise. Observe the following (i) $\mathbf{Prob}(\mathrm{Y}_{j_i} = 0) = \mathbf{Prob}(\mathrm{Y}_{j_i} = 1) = \frac{1}{2}$ for $i = 1,\ldots,s$ and (ii) the sum $\mathrm{Y} = \sum_{i=1}^{s} \mathrm{Y}_{j_i}$ of these independent variables is the number of matches between $arb_{t'}$ and the binary string $\mathrm{X}_{t'}$. Due to the Chernoff Bounds for any $0 < a < s/2$,

$$\mathbf{Prob}[|\mathrm{Y} - s/2| \geq a] \leq 2e^{-2a^2/3s},$$

For the choices of $s = p$ and $a = \frac{\sqrt{p\lambda}}{16}$, the first condition holds with at most probability $2e^{\frac{-\lambda}{384}} = 2\epsilon^{1/384}$.

Similarly, for the choices of $s = d$ and $a = \frac{\sqrt{d\lambda}}{8}$, the second condition holds with at most probability $2e^{\frac{-\lambda}{96}} = 2\epsilon^{1/96}$.

Hence, a successful framing can happen with probability at most $4\epsilon^{1/384}$.

4.2 The Proof of Theorem 2

Before we proceed with the proof of the Theorem, we first recall Chor-Fiat-Naor secret fingerprinting code based on the parameters of [12]: for an alphabet size of $2w$, where w is the size of the traitor coalition, and a length of $\ell_1 = 4w \log\left(\frac{n}{\varepsilon_1}\right)$, the failure probability of identification is bounded by ε_1. The codewords are generated totally in a random fashion by assigning a random symbol from the alphabet for each codeword position. Upon observing a pirate codeword, the number of matches between each original codeword is computed, the maximum match (i.e. score) will identify a traitor. The rationale of the accusation lies on the fact that $4\log\left(\frac{n}{\varepsilon_1}\right)$ number of matches is unlikely while on the other hand a traitor coalition of w will, regardless of the pirate strategy, lead one of them to have a score of such number.

We concatenate our asymmetric fingerprinting code (as an inner code) with a w-identifier Chor-Fiat-Naor fingerprinting code[5] (as an outer code). To achieve the parameters of the theorem we set $\varepsilon_1 = \varepsilon/2$ and $\varepsilon_2 = \frac{\varepsilon}{2\ell_1}$ in the following description the concatenation algorithms.

We next describe the resulting algorithms of the concatenation.

Code Generation: We first generate an $(\ell_1, n, 2w)$-code from secret Chor-Fiat-Naor code where the symbols of the code is chosen from the set $\{1,2,\ldots,2w\}$. We call this code by outer code. Each user is associated with a codeword from the outer code. This codeword is called to be the outer codeword of that user. More specifically, we assign the k-th codeword to the k-th user.

We choose ℓ_1 many permutations, randomly, over the set $\{1,2,\ldots,2wd\}$ where $d \stackrel{def}{=} 8(2w)^2 \ln 1/\varepsilon_2$ and $\ell_2 \stackrel{def}{=} 2wd$. Let us denote the permutations by $\pi_1,\ldots,\pi_{\ell_1}$. For each permutation π_j, we build private-input free version of the master matrices M_{d,π_j} of Figure 1 with $2w$ many rows. When we say private-input free version, we mean the blocks with private-inputs are left empty at this stage. Let us call these blocks by empty-slots.

Let $\langle c_1,\ldots,c_{\ell_1}\rangle$ be the outer codeword of a user U, and let $X^U \in \{0,1\}^d$ be the private input of user U. We generate the overall codeword of that user as follows:

For each $0 < j \leq \ell_1$, the private input X^U is placed in the empty-slot of the c_j-th row of the master matrix M_{d,π_j} as our code suggests in Section 2. Denoting this row as u_j: the resulting codeword of user U after concatenation would be $\langle u_1,\ldots,u_{\ell_1}\rangle$. Let us name these blocks at the outer level by outer-blocks, i.e. the j-th outer-block of U's codeword would be u_j.

Each receiver plugs its private input, that is of length $d = 8(2w)^2 \ln 1/\varepsilon_2$, into each symbol of its outer codeword. In other words, the same private input X is repeatedly used ℓ_1 times for each symbol of the outer codeword. The resulting code after concatenation would be a $(\ell_1 \cdot \ell_2, n, 2)$-code.

Identification: Upon obtaining a pirate codeword P we first chop the pirate codeword into ℓ_1 outer-blocks, denoted by $P = \langle p_1,\ldots,p_{\ell_1}\rangle$. Each outer-block of the pirate codeword corresponds to a pirate codeword at the inner code level. Hence, we run, in parallel, the identification algorithm of the inner code(i.e. our asymmetric code of Section 2) for each outer block: more specifically for the inputs p_i and π_i, the identification algorithm returns an index $h_i \in [2w]$ and produces $\Omega_i = (arb_i, b_i)$ as an arbiter proof.

After the above stage, we obtain a vector $\langle h_1, h_2,\ldots,h_{\ell_1}\rangle$ as a pirate codeword at the outer code level. We run the identification algorithm of the outer code which outputs a traitor index $h \in [n]$. This accused traitor, due to the correctness of the outer code, is responsible of piracy in ℓ_1/w many outer-blocks the pirate codeword P. Let us denote the indices of those outer-blocks by by $\{r_1,\ldots,r_{\ell_1/w}\} \subseteq [\ell_1]$. The identification procedure of the concatenated code will accuse the receiver with index $h \in [n]$ as a traitor and output $\Omega = \{\Omega_{r_i}\}_{i\in[\ell_1/w]}$ as the arbiter proof.

ArbiterPredicate: Upon receiving $(tk, \{X^j\}_{j\in[n]}, h, \Omega)$ where tk is the tracing key (the secret information generated at the code generation phase due to both inner and outer codes), we run, in parallel, the arbitration algorithm of the inner code(i.e. our asymmetric code of Section 2) for each sub-arbiter proof Ω_{r_i}: more specifically we run the arbiter predicate with input $(\pi_{r_i}, \{X^j\}_{j\in[n]}, h, \Omega_{r_i})$. This essentially, makes a comparison between the private input X^h of the h-th user with the string (arb_{r_i}). If at least one of these protocol-runs outputs true then we also return true.

We finally provide a sketch of the analysis for the above concatenation with the selected choices:

(i) Traceability: the correctness of the traceability merely relies on the correctness of the underlying codes. Hence, the failure probability will be bounded by $2\ell_1\varepsilon_2 + \varepsilon_1$.

(ii) Non-repudiation: Based on the identification algorithms, the accused traitor with index h has found to have

at least ℓ_1/w many matchings with the pirate codeword at the outer level. However, this does not necessarily mean that the traitor was actively using its private input in all of these ℓ_1/w affiliated pirate inner codewords. Indeed, a number of other traitors, with the same symbol at the outer level, might also actively use their private inputs (even to the extent that the accused traitor may not be active for that symbol). For such arbiter proof, the arbiter predicate will probably return false. What we need is at least one position where the accused traitor found to be single in having its symbol, i.e. all other traitors get a different symbol (otherwise we say a collusion of type h occurs). We now claim that finding a position without type h collusion is very likely given the parameters of the concatenation.

For one particular outer-block, the probability of the existence of a traitor having the same symbol what the h-th user (who has been accused of being traitor) has would be at most $\frac{w-1}{2w} < 1/2$. This is indeed because the outer-codewords are chosen randomly for our choice of $2w$ alphabet size where w is the size of the traitor coalition. Hence the probability that there are collisions of type h at all ℓ_1/w positions is bounded by $(1/2)^{4\log n/\varepsilon_1} = (\frac{\varepsilon_1}{n})^4$.

For such position where the traitor with index h obtains a different symbol from the rest of the traitors, the arbiter predicate will return true, i.e. the correlation of the arbiter string to the private input of h-th receiver will be suffice to accuse, with a non-repudiation error of at most $\varepsilon_2^{1/112}$. Hence, the overall non-repudiation error is bounded by $(\frac{\varepsilon_1}{n})^4 + \varepsilon_2^{1/112}$.

(iii) Non-framing: The ArbiterPredicate is provided $\ell_1/w = 4\log n/\varepsilon_1$ many guesses for the private input of the accused traitor. Hence, the probability of a framing success would be bounded by $4\log n/\varepsilon_1$ many times of the single framing success. Hence, a malicious code designer will be able to frame with at most $4\frac{\ell_1}{w}\varepsilon_2^{1/384}$ probability.

5. ACKNOWLEDGMENTS

This research is conducted in part while at Nanyang Technological University; supported by The Singapore National Research Foundation under Research Grant NRF-CRP2-2007-03. The author thanks to Prof. San Ling and Prof. Huaxiong Wang for the fruitful discussions.

6. REFERENCES

[1] I. Biehl, B. Meyer: Protocols for Collusion-Secure Asymmetric Fingerprinting STACS 1997: 399-412

[2] D. Boneh and M. Naor: Traitor tracing with constant size ciphertext. In proceedings of the 15th ACM conference on Computer and Communications Security (CCS '08), pp. 501–510, 2008.

[3] D. Boneh, J. Shaw: Collusion-Secure Fingerprinting for Digital Data (Extended Abstract). CRYPTO 1995

[4] A. Charpentier, C. Fontaine, T. Furon, I. J. Cox: An Asymmetric Fingerprinting Scheme based on Tardos Codes. Information Hiding 2011: 43-58

[5] B. Chor, A. Fiat, M. Naor: Tracing Traitors. CRYPTO 1994: 257-270

[6] S. Craver, N. Memon, B. Yeo, and M. M. Yeung. Resolving rightful ownerships with invisible watermarking techniques: Limitations, attacks, and implications. IEEE Journal on Selected Areas in Communications, 16(4):573-586, May 1998.

[7] M. Deng, T. Bianchi, A. Piva and B. Preneel: An Efficient Buyer-Seller Watermarking Protocol based on composite signal representation. 11th ACM Workshop on Multimedia and Security, Princeton, NJ, 2009, pp. 9-18

[8] M. Deng, L. Weng, B. Preneel: Anonymous Buyer-Seller Watermarking Protocol with Additive Homomorphism. SIGMAP 2008: 300-307

[9] Ju, H. S., Kim, H. J., Lee, D. H., and Lim, J. I. (2002). An anonymous buyer-seller watermarking protocol with anonymity control. Information security and cryptology ICISC 2002

[10] H. Jin, J. Lotspiech: Renewable Traitor Tracing: A Trace-Revoke-Trace System For Anonymous Attack. ESORICS 2007: 563-577

[11] A. Kiayias, S. Pehlivanoglu: Tracing and Revoking Pirate Rebroadcasts. ACNS 2009: 253-271

[12] A. Kiayias and S. Pehlivanoglu. Encryption for Digital Content, volume 52 of Advances in Information Security. Springer, 2010.

[13] A. Kiayias, S. Pehlivanoglu: Improving the Round Complexity of Traitor Tracing Schemes. ACNS 2010

[14] M. Kuribayashi, H. Tanaka: Fingerprinting protocol for images based on additive homomorphic property. IEEE Transactions on Image Processing 14(12), 2005

[15] A. Kiayias and M. Yung, On Crafty Pirates and Foxy Tracers, ACM CCS-8 Workshop DRM 2001

[16] N. D. Memon, P. W. Wong: A buyer-seller watermarking protocol. IEEE Transactions on Image Processing 10(4): 643-649 (2001)

[17] J. P. Prins, Z. Erkin, R. L. Lagendijk: Anonymous Fingerprinting with Robust QIM Watermarking Techniques. EURASIP J. Information Security 2007

[18] B. Pfitzmann, M. Schunter: Asymmetric Fingerprinting. EUROCRYPT 1996: 84-95

[19] B. Pfitzmann, M. Waidner: Asymmetric Fingerprinting for Larger Collusions. ACM Conference on Computer and Communications Security 1997

[20] B. Pfitzmann, M. Waidner: Anonymous Fingerprinting. EUROCRYPT 1997: 88-102

[21] Min-Hua Shao. A privacy-preserving buyer-seller watermarking protocol with semi- trust third party. In Trust, Privacy and Security in Digital Business, 2007

[22] J. Staddon, D. R. Stinson, R. Wei: Combinatorial properties of frameproof and traceability codes. IEEE Transactions on Information Theory 47(3) 2001

[23] D. R. Stinson, R. Wei: Combinatorial Properties and Constructions of Traceability Schemes and Frameproof Codes. SIAM J. Discrete Math. 11(1): 41-53 (1998)

[24] R. Safavi-Naini, Y. Wang: New results on frame-proof codes and traceability schemes. IEEE Transactions on Information Theory 47(7): 3029-3033 (2001)

[25] Z. Jane Wang, Min Wu, H. Vicky Zhao, Wade Trappe, K. J. Ray Liu: Anti-collusion forensics of multimedia fingerprinting using orthogonal modulation. IEEE Transactions on Image Processing 14(6), 2005

[26] Gabor Tardos: Optimal probabilistic fingerprint codes. J. ACM 55(2): (2008)

[27] J. Zhang, W. Kou, and K. Fan. Secure buyer-seller watermarking protocol. IEE Proceedings Information Security, 153(1):15-18, March 2006.