An Asymmetric Fingerprinting Scheme based on Tardos Codes

Ana Charpentier, Caroline Fontaine, Teddy Furon, Ingemar Cox

▶ **To cite this version:**

Ana Charpentier, Caroline Fontaine, Teddy Furon, Ingemar Cox. An Asymmetric Fingerprinting Scheme based on Tardos Codes. Information Hiding, May 2011, Prague, Czech Republic. inria-00581156v1

HAL Id: inria-00581156

https://inria.hal.science/inria-00581156v1

Submitted on 26 Sep 2013 (v1), last revised 24 Apr 2017 (v2)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

An Asymmetric Fingerprinting Scheme based on Tardos Codes

Ana Charpentier[(a)], Caroline Fontaine[(b)], Teddy Furon[(a)], Ingemar Cox[(c)]

[(a)]INRIA-Rennes research center, Campus de Beaulieu, Rennes, France
[(b)]CNRS/Lab-STICC/CID, Télécom Bretagne/ITI, Brest, France
[(c)]University College London, Dpt. of Computer Science, London, United Kingdom

Abstract. Tardos codes are currently the state-of-the-art in the design of practical collusion-resistant fingerprinting codes. They rely on a secret vector drawn from a publicly known probability distribution in order to generate each Buyer's fingerprint. For security purposes, this secret vector must not be revealed to the Buyers. To prevent an untrustworthy Provider forging a copy of a Work with an innocent Buyer's fingerprint, previous asymmetric fingerprinting algorithms enforce the idea of the Buyers generating their own fingerprint. Applying this concept to Tardos codes is challenging since the fingerprint must be based on this vector secret.
This paper provides the first solution for an asymmetric fingerprinting protocol dedicated to Tardos codes. The motivations come from a new attack, in which an untrustworthy Provider by modifying his secret vector frames an innocent Buyer.

1 Introduction

This paper considers a problem arising in the fingerprinting of digital content. In this context, a fingerprint is a binary code that is inserted into the Work for the purpose of protecting it from unauthorized use, or, more precisely, for the purpose of identifying individuals responsible for unauthorized use of a Work. In such a scenario, it is assumed that two or more users may collude in order to try to hide their identities. Under the *marking assumption* [1], colluders cannot alter those bits of the code that are identical for all colluders. However, where bits differ across colluders, these bits may be assigned arbitrary values. A key problem is resistance to collusion, i.e. if c users create a pirated copy of the Work, its tampered fingerprint (i) should not implicate innocent users, and (ii) should identify at least one of the colluders.

This problem has received considerable attention since Boneh and Shaw [1] discussed the problem. They introduced the concept of a c-secure code such that the probability of framing an innocent is lower than ϵ. Unfortunately, the length of their codes, $O(c^4 \log(\frac{n}{\epsilon}) \log(\frac{1}{\epsilon}))$ where n is the number of users, was too long to be practical. Following Boneh and Shaw's paper, there has been considerable effort to design shorter codes. In 2003, Tardos [2] proposed an efficient code construction that, for the first time, reduced the code length to the lower bound,

$O(c^2 \log(\frac{n}{\epsilon}))$, thereby making such codes practical. Tardos codes are currently the state-of-the-art for collusion-resistant fingerprinting.

Meanwhile, several papers have considered the scenario where the Provider is untrustworthy. Thanks to the knowledge of a Buyer's fingerprint, the Provider creates a pirated copy of a Work, implicating this innocent Buyer. To prevent this, Pfitzman and Schunter [3] first introduced the concept of asymmetric fingerprinting in which the Provider doesn't need to know the Buyer's fingerprint. The Buyer first commits to a secret (the fingerprint) that only he/she knows. The Buyer and Provider then follow a protocol which results in the Buyer receiving a copy of the Work with his/her secret fingerprint (and some additional information coming from the Provider) embedded within it. The Provider did not learn the Buyer's secret, and cannot therefore create a forgery. Unfortunately, the early implementations of this concept were not practical due to the huge code length. Moreover, these previous asymmetric fingerprinting methods cannot be applied to much more efficient Tardos codes because the fingerprints must be drawn from a particular probability distribution depending on a secret vector only known to the Provider.

Sec. 2.2 first shows some vulnerabilities of Tardos decoding, especially when the Provider is not trusted. Given any unauthorized copy, i.e. a Work that does *not* contain the innocent Buyer's fingerprint, the Provider can alter its secret vector in order to accuse an arbitrary Buyer. This motivates a new asymmetric fingerprinting protocol dedicated to the use of Tardos codes and presented in Sec. 3, preventing prevent both the Buyer and the Provider from cheating. Sec. 4 then discusses practical aspects of the fingerprints embedding and accusation. We finally discuss our solution in Sec. 6 before concluding.

2 Untrustworthy Provider with the Tardos code

For readers unfamiliar with Tardos codes, we now provide a brief introduction. Further details can be found in [4].

2.1 Tardos code in a nutshell

Let n denote the number of buyers, and m the length of the code. The fingerprints can then be arranged as a binary $n \times m$ matrix $\mathbf{X}$, Buyer j being related to the binary fingeprint $\mathbf{X}_j = (X_{j1}, X_{j2}, \ldots, X_{jm})$.

To generate this matrix, m real numbers $p_i \in [t, 1-t]$ are generated, each of them being randomly and independently drawn according to the probability density function $f : [t, 1-t] \to \mathbb{R}^+$ with $f(z) = \kappa(t)(z(1-z))^{-1/2}$ and $\kappa(t)^{-1} = \int_t^{1-t}(z(1-z))^{-1/2}dz$. The parameter $t \ll 1$ is referred to as the cutoff. We set $\mathbf{p} = (p_1, \ldots, p_m)$. This vector $\mathbf{p}$ is the secret key of the code only known by the Provider. Each element of the matrix $\mathbf{X}$ is then independently randomly drawn, such that the probability that the element X_{ji} is set to symbol '1' is $\mathbb{P}(X_{ji} = 1) = p_i$. The fingerprint $\mathbf{X}_j$ is then embedded into the copy of the Work thanks to a watermarking technique, before being accessed by Buyer j.

When an unauthorized copy is found, a sequence $\mathbf{Y}$ is extracted thanks to the watermarking decoder. Due to collusion and possible distortions such as transcoding, this sequence is unlikely to exactly equal one of the fingerprints in the matrix $\mathbf{X}$. To determine if Buyer j is involved in the production of the unauthorized copy, a score, referred to as an accusation score, S_j is computed. If this score is greater than a given threshold Z, then Buyer j is considered to have colluded. The value of threshold Z theoretically guarantees that the probability of accusing an innocent is below the significance level ϵ.

The scores are computed according to an accusation function g, reflecting the impact of the correlation between the fingerprint $\mathbf{X}_j$, associated with Buyer j, and the decoded sequence $\mathbf{Y}$:

$$S_j = G(\mathbf{Y}, \mathbf{X}_j, \mathbf{p}) = \sum_{i=1}^{m} g(Y_i, X_{ji}, p_i). \tag{1}$$

In the usual symmetric codes [4], function g is constrained (such that, for example, for an innocent, the expectation of the score is zero and its variance is m), giving $g(1,1,p) = g(0,0,1-p) = -g(0,1,p) = -g(1,0,1-p) = \sqrt{\frac{1-p}{p}}$.

2.2 Untrustworthy content provider

We now consider the case where the Provider is no longer trusted, and, as such, wishes to frame Buyer j for some reasons. To the unconvinced reader, we wish to outline that any caught colluder can bring this argument in court, and that this scenario is the basics of a long tradition of asymmetric fingerprinting. We further assume that the Provider has no prior access to an unauthorized copy, i.e. the Provider cannot insert a false fingerprint into the unauthorized copy, nor can he/she place a Buyer's copy on an unauthorized location. On receipt of an unauthorized copy, the untrustworthy Provider extracts the sequence present in the unauthorized copy. The hypothesis is that the underlying watermarking algorithm comes from a technology provider and that the Provider doesn't master or has no access to this technology brick. Therefore, he cannot cheat on this step of the process. Given the extracted sequence $\mathbf{Y}$, the Provider must now compare it to all known Buyers' fingerprints. This comparison is performed using Eq. (1). And it is here that the Provider can lie, since the probabilities, $\mathbf{p}$, are only known by the Provider.

An untrustworthy Provider can create a fake vector of probabilities, $\hat{\mathbf{p}}$, that implicates Buyer j. However, distribution $f(p)$, is publicly known, so the question becomes how to generate a $\hat{\mathbf{p}}$ that (i) implicates Buyer j, and (ii) has an arbitrarily high probability of been drawn from the distribution $f(p)$?

The following method shows that it is indeed extremely simple to do so. However, we do not claim that this attack is unique or optimal. Let us focus on a column where $p_i = p$ and $Y_i = X_{j,i}$. The true summand in Eq. (1) is $g(1,1,p)$ or $g(0,0,p)$ (with equal probability). Suppose that the content provider replaces the secret value p by a fake secret $\hat{p}$ which is drawn independently according to

f. On average, this summand takes the new value:

$$\Delta(t) = \int_t^{1-t} f(\hat{p}) \frac{g(1,1,\hat{p}) + g(0,0,\hat{p})}{2} d\hat{p} = \frac{1}{\pi} \ln \frac{1-t}{t}.$$

For a cutoff $t = 1/900$ (recommended by G. Tardos to fight against 3 colluders), the numerical value is surprisingly high: $\Delta(1/900) \approx 2.16$. Suppose now that the content provider applies the same strategy on an index i where $Y_i \neq X_{j,i}$. Then the expectation is the opposite. However, in a Tardos code, even for an innocent Buyer j, the proportion α of indices where symbols Y_i and $X_{j,i}$ agree is above $1/2$ for most of the collusion strategy. For instance, with an interleaving collusion attack, $\alpha = 3/4$ whatever the collusion size c.

Based on this fact, we propose the following attack. The Provider computes the score for all Buyers, which on average equals 0 for innocent Buyers and $2m/c\pi$ for the colluders [4]. The provider initializes $\hat{\mathbf{p}} = \mathbf{p}$. Then, he/she randomly selects a column i and randomly draws a fake secret $\hat{p}_i \sim f$. He/She re-computes the score of Buyer j with this fake secret and iterates selecting a different column until S_j is above the threshold Z. On average, $m(c\pi\Delta(t)(\alpha - 1/2))^{-1}$ secret values p_i need to be changed in this way, e.g. only 20% of the code length if the copy has been made using an interleaving attack.

Fig. 1 illustrates this attack for the case where the code length is $m = 1000$ and the number of colluders is $c = 3$. The solid coloured lines depict the accusation scores of 10 randomly selected innocent buyers. We observe that after between 20-30% of the elements of $\mathbf{p}$ have been altered, the accusation scores of the innocent Buyers exceed the *original* scores of the colluders. In fact, the colluders accusation scores also increase. However, we are not concerned by the highest score, but rather by the fact that the Provider is able to exhibit a couple $(\hat{\mathbf{p}}, \mathbf{X}_j)$ such that $S_j > Z$. Thus, it is sufficient to raise the score of the innocent Buyer, even if this raises all other Buyers' scores as well.

Randomly selecting some p_i's (independently from $\mathbf{X}_j$ and $\mathbf{Y}$) and re-drawing them according to the same law ensures that $\hat{p}_i \sim f$, $\forall i$. Therefore, a judge observing $\hat{\mathbf{p}}$ cannot distinguish the forgery. For this reason, the judge might request to see the matrix $\mathbf{X}$ to statistically test whether the elements of $\mathbf{X}$ are drawn from the distribution $\hat{\mathbf{p}}$. In this case, the Provider can give a fake matrix $\hat{\mathbf{X}}$ where the columns whose p_i have been modified are re-drawn such that $\mathbb{P}(X_{ki} = 1) = \hat{p}_i$, $\forall k \neq j$. The only way to prevent this deception would be that the judge randomly asked an innocent User $k \neq j$ for his copy in order to verify the authenticity of $\hat{\mathbf{X}}$. This latter step seems somewhat odd. We arrive at the silly situation where the judge has to contact an innocent Buyer k before accusing Buyer j.

3 An asymmetric Tardos code construction

In previous asymmetric fingerprinting schemes, it is up to the Buyer to generate his or her fingerprint. The Buyer then sends a commitment to the Provider, which prevents the Buyer from changing the fingerprint during the protocol.

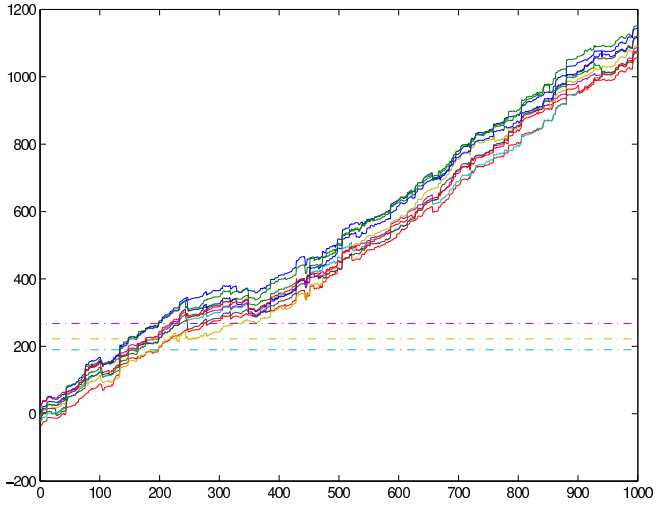

Fig. 1. Accusation score as a function of the number of changed elements of the vector $\mathbf{p}$ for the case where $m = 1000$ and $c = 3$. The solid coloured lines show how the accusation score of 10 randomly selected innocent buyers increases. The dotted horizontal lines show the original scores for the colluders before the modification.

Unfortunately, this cannot be done with a Tardos code since the fingerprint must follow a given statistical distribution controlled by $\mathbf{p}$, and $\mathbf{p}$ is only known to the Provider. This section proposes a solution to this problem, which consists of two phases. We first review its main building blocks.

3.1 Pick a card, any card!

What we need is a scheme that enables a receiver $\mathbf{R}$ to pick k elements at random in a list of N elements provided by a sender $\mathbf{S}$, in such a way that:

1. $\mathbf{R}$ gets elements that are really belonging to the list;
2. $\mathbf{R}$ does not get any information on the elements he did not pick;
3. $\mathbf{S}$ does not know which elements have been picked.

There are mainly two approaches for designing such schemes in the literature. The first one, *Oblivious Transfer protocols (OT)*, has been introduced by cryptographers in [5] and led to a huge amount of papers, *e.g.* [6–8]. The other one is mentioned in security papers as the concept of as an application of? *Commutative Encryption* or *Two-lock Cryptosystem* [9–13]. Functionally speaking, both fulfill our requirements, but they lead to different complexities and security levels (definitions and models). We find it can be interesting here to compare the two approaches, because the particular Oblivious Transfer *Two-lock Cryptosystem* adapts very well to our solution. Je ne comprends pas cette phrase.

Oblivious Transfer protocols. These protocols are studied in the same framework as multi-party computation. Let us denote by OT_k^N a k-out-of-N Oblivious Transfer protocol. In the literature we can find OT_1^2, OT_1^N and OT_k^N protocols. When $k \geq 1$, we speak of *adaptive OT protocols*, denoted by $OT_{k\times 1}^N$, if the k elements are picked one-by-one adaptively, and of *non-adaptive OT protocols* OT_k^N if they are picked simultaneously. OT security is studied under different models below listed from the weakest to the strongest: honest-but-curious model (where no one cheats during the protocol execution), half simulation (introduced by [14], cheating sender or cheating receiver studied separately; local security study), full simulation (introduced in [15], studying cheating sender and receiver globally; global security study). In addition, the UC model C'est quoi UC? has been introduced in [16] to study the behavior and security of protocols that are based on concurrent and composable cryptographic primitives.

Camenish et al. stated that the protocols presented in [14] are secure under half-simulation, and vulnerable to selective failure attacks (but can be improved to resist this attack) [15]. The scheme of [17] is secure under half simulation. Naor and Pinkas [6, 18] were the first to study the adaptive case $OT_{k\times 1}^N$. The security IND-CPA (Chosen Plaintext attack) is reached in [8]. Pourquoi IND-CPA n'est pas dans la liste des modeles ?

Commutative Encryption. An encryption scheme **CE** is said to be a *commutative encryption* if for any two set of keys k_R and k_S and any plaintext m, we have (usual definition in the literature)

$$\mathbf{CE}(k_R, \mathbf{CE}(k_S, m)) = \mathbf{CE}(k_S, \mathbf{CE}(k_R, m)). \tag{2}$$

Based on such a primitive, a *Commutative Encryption Scheme (CES)* can be defined as follows [9].

1. Let m_1, m_2, ...,m_N the input of the Sender **S**. **S** chooses N secret keys K_1, K_2, ..., K_N for a symmetric cryptosystem **E** (*e.g.* AES, DES) and a key k_S for the commutative encryption primitive **CE**. Let

$$\begin{aligned} C_1 &= \mathbf{E}(K_1, m_1)\ , & D_1 &= \mathbf{CE}(k_S, K_1) \\ C_2 &= \mathbf{E}(K_2, m_2)\ , & D_2 &= \mathbf{CE}(k_S, K_2) \\ &\cdots & &\cdots \\ C_N &= \mathbf{E}(K_N, m_N)\ , & D_N &= \mathbf{CE}(k_S, K_N) \end{aligned}$$

   Couples $\langle C_j,\ D_j\rangle$ can be publicly accessed.

2. Assume that the receiver **R** wants m_i. **R** loads $\langle C_i,\ D_i\rangle$ and chooses a secret key k_R for **CE**. He encrypts D_i with it and sends the result $U = \mathbf{CE}(k_R, D_i)$ to **S**.
3. **S** decrypts U with S and sends $W = \mathbf{CE}^{-1}(k_S, U)$ to **R**. **R** computes K_i, and can get to $m_i = \mathbf{E}^{-1}(K_i, C_i)$. The list of encrypted items C_i is needed, but it would be better to have commitments instead of symmetric encryption **E**. Cette phrase n'est pas claire.

Let us discuss now about the choice of primitive **CE** appropriate for this protocol. **CE** is always used in a "symmetric" way, that is, encryption and decryption are both performed by either **S** or **R**. Hence using symmetric or asymmetric encryption schemes does not matter, functionally speaking. What is more important is the security level we want to achieve. Here we cannot achieve unconditional security: it is not possible to use a One-Time Pad because the same key k_S encrypts all the keys K_i. Hence, semantic security is the best security class we might achieve. Nevertheless, the practical schemes proposed in the literature [9, 13, 11] are not semantically secure. The authors seem to have noticed that and have added few lines to say that it can be made semantically secure by adding some padding format.

But semantic security is very important in our case, because we have to encrypt binary symbols and do not want the Receiver to be able to distinguish encrypted 0's from encrypted 1's, during the fingerprint generation or the half-word disclosure steps. This implies the use of a probabilistic encryption scheme. Although it is never presented like this in the literature, we find more rigorous to rewrite the CE property of (2) as

$$\mathbf{CE}^{-1}(k_R^*, \mathbf{CE}^{-1}(k_S^*, \mathbf{CE}(k_R, \mathbf{CE}(k_S, m)))) = m, \tag{3}$$

where k^* denotes the decryption key associated to k of a probabilistic encryption scheme. An example of probabilistic asymmetric commutative encryption scheme achieving semantic security is ElGamal, which is IND-CPA. Up to our knowledge, there are no IND-CCA1 (and then no IND-CCA2) commutative encryption schemes[1]

A few practical implementations of a two-lock cryptosystem have been proposed: a first one based on the Knapsack problem [11], which has been broken [12], a second one based on the discrete logarithm problem [11], and a third one based on RSA [13]. For more details on the mentioned security classes or encryption schemes, we will refer to [19–21], as it would be too long to recall them in the article.

Which Oblivious Transfer scheme ? The OT scheme build on Commutative Encryption fits very well with what we want to do, but we haven't found in the literature a practical scheme whose security have been as studied and tested as the other OT schemes. Then we propose a description using a slightly modified Commutative Encryption Scheme and a description using any OT scheme.

3.2 Phase 1: Generation of the fingerprint

Commutative Encryption. We use the above commutative encryption protocol m times to generate the fingerprint of the j-th Buyer $\mathbf{X}_j = (X_{j,1}, \ldots, X_{j,m})$. **S**

[1] Some may be derived from IND-CCA1 and IND-CCA2 usual schemes, as Cramer-Shoup for example, but it is a complete research challenge to define them properly and prove their security as such a proof would need several oracles, which is not the case for the usual schemes. It is out of the scope of our work here.

is the Provider, and $\mathbf{R}$ is Buyer j. The Provider generates a secret vector $\mathbf{p}$ for a Tardos code. Each p_i is quantized such that $p_i = L_i/N$ with $L_i \in [N-1]$.

For a given index i, the objects are the concatenation of a binary symbol and a text string. There are only two versions of an object in list C_i. For L_i objects, $O_{k,i} = (1\|\mathtt{ref}_{1,i})$, and $O_{k,i} = (0\|\mathtt{ref}_{0,i})$ for the $N - L_i$ remaining ones. The use of the text strings $\{\mathtt{ref}_{X,i}\}$ depends on the content distribution mode as detailed in Sec. 4.2. The object $O_{k,i}$ is committed with key $K_{k,i}$ and stored in the list $\mathcal{C}_i = \{C_{k,i}\}_{k=1}^N$. There are thus as many different lists $\mathcal{C}_i$ as the length m of the fingerprint. These lists are published in a public Write Once Read Many (WORM) directory [22] whose access is granted to all users. As explicitly stated in its name, nobody can modify or erase what has been put the first time in a WORM directory; beside, anybody can check its integrity. This list is the C_i list in the Commutative Encryption (Sec. 3.1), and have to be added in the use of another OT scheme.fin de phrase pas claire.

$$
\begin{array}{lcll}
p_1 & \overset{quantization}{\longrightarrow} & (0\|\mathtt{ref}_{0,1}, 1\|\mathtt{ref}_{1,1}, \ldots, 1\|\mathtt{ref}_{1,1}) & \overset{Commitment}{\longrightarrow} \quad \mathcal{C}_1 = (C_{1,1}, \ldots, C_{N,1}) \\
p_2 & \longrightarrow & (0\|\mathtt{ref}_{0,2}, 0\|\mathtt{ref}_{0,2}, \ldots, 0\|\mathtt{ref}_{0,2}) & \longrightarrow \quad \mathcal{C}_2 = (C_{1,2}, \ldots, C_{N,2}) \\
 & & \vdots & \\
p_m & \longrightarrow & (1\|\mathtt{ref}_{1,m}, 0\|\mathtt{ref}_{0,m}, \ldots, 1\|\mathtt{ref}_{1,m}) & \longrightarrow \quad \mathcal{C}_m = (C_{1,m}, \ldots, C_{N,m})
\end{array}
$$

Fig. 2. The lists $\mathcal{C}_i = \{C_{k,i}\}_{k=1}^N$ are stored in a WORM

On the contrary, the $\mathcal{D}$-lists are made specific to a given Buyer j. The Provider picks a secret key S_j and a permutation $\pi_j(.)$ over $[N]$. This Buyer is proposed a list $\mathcal{D}_{j,i} = \{D_{j,i,k} = \mathbf{CE}(S_j, (\pi_j(k)\|K_{\pi_j(k),i}))\}_{k=1}^N$. Therefore, the lists $\{\mathcal{C}_i\}_{i=1}^m$ are common for all users, whereas the lists $\{\mathcal{D}_{j,i}\}_{i=1}^m$ are specific to Buyer j. We have introduced here a slight change with respect to protocol 3.1, i.e. the permutation π_j whose role is explained below. Buyer j chooses one object in the list, say the $k(j,i)$-th object. He/she sends the corresponding ciphertext $U_{k(j,i),i} = \mathbf{CE}(R_{j,i}, D_{j,i,k(j,i)})$ decrypted by the provider with S_j and sent back to the Buyer who, at the end, gets the index $\mathsf{ind}(j,i) = \pi_j(k(j,i))$ and the key $K_{\mathsf{ind}(j,i),i}$, which grants him/her the access to the object $O_{\mathsf{ind}(j,i),i}$, store encrypted in the WORM. It contains the symbol $b_{\mathsf{ind}(j,i),i}$. This will be the value of the i-th bit of his/her fingerprint, $X_{j,i} = b_{\mathsf{ind}(i,j),i}$, which equals '1' with probability p_i.

Oblivious Transfer protocols. Buyer and Provider run an Oblivious Transfer to get the corresponding key $K_{\mathsf{ind}(j,i),i}$: the Provider proposes the list of the keys $\{\pi_j(k)\|K_{\pi_j(k),i}\}$ and the Buyer picks one with an OT_1^N. This key allows him to open one of the commitments $C_{\pi_j(k),i}$.

The provider keeps in a log file the values of S_j and $U_{k(j,i),i}$, the user keeps $R_{j,i}$ in his/her records. Ne marche qu'avec CE. pas de U et R avec OT.

	P		B
1.	**Commitment** $C_{\pi_j(k),i} = \mathbf{Com}(K_{\pi_j(k),i}, O_{\pi_j(k),i})$ Only one time for every users and every bits		
2.	$D_{j,i,k} = \mathbf{CE}(S_j, (\pi_j(k) \| K_{\pi_j(k),i}))$		
3.		$\longleftarrow$	$U_{k(j,i),i} = \mathbf{CE}(R_{j,i}, D_{j,i,k(j,i)})$
4.	$V = \mathbf{CE}^{-1}(S_j, U_{k(j,i),i})$	$\longrightarrow$	
5.			$\pi_j(k) \| K_{\pi_j(k),i} = \mathbf{CE}^{-1}(R_{j,i}, V)$
6.			$O_{\pi_j(k),i} = \mathbf{E}^{-1}(K_{\pi_j(k),i}, C_{\pi_j(k),i})$

Fig. 3. Generation of a fingerprint bit. C'est le schma pour OT ou pour CE ?

3.3 Phase 2: Disclosure of the halfword

The practical version of the accusation process detailed in Sec. 4.3 allows the Provider to list a bunch of suspected users to be forwarded to the judge for verification. After phase 1 is completed, the Provider orders Buyer j to reveal $m_h < m$ bits of his fingerprint. These disclosed symbols compose the so-called halfword [3]. The following facts must be enforced: Buyer j doesn't know which bits of his/her fingerprint are disclosed even if the Provider asks for the same bit indices to all the users. The Provider discloses m_h bits of the fingerprints without revealing any knowledge about the others. Of course, Buyer j refuses to follow the protocol for more than m_h objects.

Commutative Encryption. Again, we propose to use the double-blind random selection protocol of Sec. 3.1. Now, Buyer j plays the role of **S**, and the Provider the role of **S**, $N = m$, and object $O_i = (R_{i,j} \| \texttt{alea}_{i,j})$. These items are the m secret keys selected by Buyer j during phase 1 (Sec. 3.2) concatenated with random strings $\texttt{alea}_{i,j}$ to be created by Buyer j. This alea finds its use during the personalization of the content (see Sec. 4.2). Following the protocol, the Provider selects m_h such object. The decryption of message $U_{k(i,j),j}$ received during phase 1 thanks to the disclosure of the key $R_{i,j}$ yields $D_{i,j,k(i,j)}$ which in turn decrypted with key S_j provides the index of the selected object, otherwise the protocol stops. This prevents a colluder from denying the symbol of his fingerprint and from copying the symbol of an accomplice. At the end, the Provider learns which item was picked by Buyer j at index i. Therefore, he/she ends up with m_h couples $(X_{j,i}, \texttt{alea}_{k(i,j),i})$ associated to a given Buyer j.

Oblivious Transfer protocols. At phase 2, any $OT^N_{k\times 1}$ can be used to allow the Provider to get m_h objects from the list of the $O_i = (R_{i,j} \| \texttt{alea}_{i,j})$ owned by the Buyer. The problem is if another OT scheme was used at the precedent step, there is no such things as the $R_{i,j}$ values. In order to prevent the Buyer from denying the symbol of his fingerprint, the $R_{i,j}$ values have to be replaced by a number which was part of the exchange during the generation of the fingerprint. This element will be specific to the OT protocol.

4 The remaining bricks of the scheme

The previous section has detailed the core of our scheme which is the construction of the codewords based on oblivious transfer. This section deals with the watermarking of video content, the distribution and the accusation process.

4.1 Watermarking

A nowadays trend is the application of fingerprinting to premium video contents. Premium means movies in very high quality available for home cinema shortly after their release in theaters. Personalization of the copies are usually done as follows: Before distribution, the content is divided into sequential blocks (e.g. Group of Pictures of few seconds of a video). Offline, a robust watermarking technique creates two versions of some blocks embedding the symbol '0' and respectively '1'. This is done by a technology provider and the copyright owner or artist can check the quality of the watermarked content before distribution by the content provider. Quality is very important for premium movies and watermarking under that constraint involves a lot of processing. This motivates this offline preprocessing.

In some scenarios (screeners for jurys, marketing, blu-ray discs, premium downloads), the physical medium storage or bandwitth is so large that both versions of the blocks are encrypted and transmitted to the software client or the device of the Buyers. This latter is trusted and the strings $\{\mathtt{ref}_{X,i}\}$ it got from phase 1 are parameters needed to get access to the i-th block watermarked with symbol X.

4.2 Content personalization at the server side

As for Video On Demand where the client is not trusted, personalization of the content is usually made at the server side, which raises an issue since the provider doesn't know user fingerprints. There exist buyer-seller protocols for embedding a sequence $\mathbf{X}_j$ into a content c_o without disclosing $\mathbf{X}_j$ to the seller and c_o to the buyer. They are based on homomorphic encryption scheme and work with some specific implementations of spread spectrum [23] or Quantization Index Modulation watermarking [24]. In other words, not any watermarking technique can be used, and this is not the route we have chosen so far. Due to space limitations, a brief sketch of the adaptation of [24] is presented hereafter.

Let $\mathbf{c}_i^{(0)} = (c_{i,1}^{(0)}, \ldots, c_{i,Q}^{(0)})$ be the Q quantized components (like pixels, DCT coefficients, portion of streams etc) of the i-th content block watermarked with symbol '0' (resp. $\mathbf{c}_i^{(1)}$ with symbol '1'). Denote $\mathbf{d}_i = \mathbf{c}_i^{(1)} - \mathbf{c}_i^{(0)}$. Assume as in [24, Sect. 5], an additive homomorphic and probabilistic encryption $E[.]$ such as the Pallier cryptosystem. Buyer j has a pair of public/private keys (pk_j, sk_j) and sends $(E_{pk_j}[X_{j,1}], \ldots, E_{pk_j}[X_{j,m}])$. The provider sends him/her the ciphers

$$E_{pk_j}[c_{i,\ell}^{(0)}].E_{pk_j}[X_{j,i}]^{d_{i,\ell}}, \forall (i,\ell) \in [m] \times [Q]. \tag{4}$$

Thanks to the homomorphism, Buyer j decrypts this with sk_j into $c_{i,\ell}^{(0)}$ if $X_{j,i} = 0$, $c_{i,\ell}^{(1)}$ if $X_{j,i} = 1$. Since $X_{j,i}$ is constant for the Q components of the i-th block, a lot of bandwidth and computer power will be saved with a composite signal representation as detailed in [24, Sect. 3.2.2].

A crucial step in this kind of buyer-seller protocols is to prove to the Provider that what is sent by the Buyer is indeed the encryption of bits, and moreover bits of the Buyer's fingerprint. This usually involves complex zero-knowledge subprotocols [23, 24]. Here, we avoid this complexity by taking advantage of the fact that the Provider already knows some bits of the fingerprint $\mathbf{X}_j$, i.e. those belonging to the halfword (see Sec. 3.3), and the Buyers do not know the indices of these bits. Therefore, in $m_v < m_h$ random indices of the halfword, the Provider asks Buyer j to open his/her commitment. For one such index i_v, Buyer j reveals the random value r_{i_v} of the probabilistic Pallier encryption (with the notation of [24]). The Provider computes $g^{X_{j,i_v}} h^{r_{i_v}} \mod N$ and verifies it equals the i_v-th cipher, which Buyer j pretended to be $E_{pk_j}[X_{j,i}]$.

One drawback of this simple verification scheme is that the Buyer discovers m_v indices of the halfword. This may give rise to more elaborated collusion attacks. For example, Buyer j, as a colluder, could try to enforce $Y_{i_v} \neq X_{j,i_v}$ when attempting to forge a pirated copy. Further discussion of this is beyond the scope of this paper.

This approach may also introduce a threat to the Buyer. An untrustworthy Provider can ask to open the commitments of non-halfword bits in order to disclose bits he/she is not supposed to know. For this reason, the Provider needs to send $\texttt{alea}_{k(i_v,j),i_v}$ as defined in Sec. 3.3 to show Buyer j that his/her verification duly occurs on a halfword bit.

4.3 The accusation procedure

The accusation is straightforward and similar to other fingerprinting protocols. A scouting agency is in charge of catching a forgery. The technology provider decodes the watermark and extracts sequence $\mathbf{Y}$ from the pirated content. The Provider computes the halfscores by applying Eq. (1) only on the halfwords. This produces a list of suspects, e.g. those users whose score is above a threshold, or those users with the highest scores.

Of course, this list cannot be trusted, since the Provider may be untrustworthy. The list is therefore sent to a third party, referred to as the Judge, who first verifies the computation of the halfscores. If different values are found, the Provider is black-listed. Otherwise, the Judge computes the scores of the full fingerprint.

To do so, the Judge needs the secret $\mathbf{p}$: he/she asks the Provider for the keys $\{K_{k,i}\}$, $\forall (k,i) \in [N] \times [m]$ and thereby obtains from the WORM all the objects $\{O_{k,i}\}$, and the true values of $(p_1, \ldots, p_m)$. The Judge must also request suspected Buyer j for the keys $R_{j,i}$ in order to decrypt the messages $U_{k(j,i),i}$ in $D_{i,j,k(i,j)}$ which reveal which object Buyer j picked during the i-th round of Sec. 3.2 and whence $X_{j,i}$. Finally, the Judge accuses the user whose score over

the full length fingerprint is above a given threshold (related to a probability of false alarm).

5 Discussion

5.1 Security

Suppose first that the Provider is honest and denote by c the collusion size. A reliable tracing capability on the halfwords is needed to avoid false alarms. Therefore, as proven by G. Tardos, $m_h = O(c^2 \log n\epsilon^{-1})$, where ϵ is the probability of suspecting some innocent Buyers. Moreover, successful collusions are avoided if there are secret values such that $p_i < c^{-1}$ or $p_i > 1 - c^{-1}$(see [25]). Therefore, N should be sufficiently big, around a hundred, to resist against collusion of size of some tens. During the generation of the fingerprint in Sec. 3.2, permutation $\pi_j(.)$ makes sure that Buyer j randomly picks up a bit '1' with probability $p_i = L_i/N$ as needed in the Tardos code. In particular, a colluder cannot benefit from the discoveries made by his accomplices.

We now analyze why colluders would cheat during the watermarking of their version of the Work described in Sec. 4.2. By comparing their fingerprints, they see indices where they all have the same symbols, be it '0' or '1'. As explained in the introduction, they won't be able to alter those bits in the tampered fingerprint except if they cheat during the watermarking: If their fingerprint bits at index i all equal '1', one of them must pretend he/she has a '0' in this position. If they succeed to do so for all these positions, they will able to forge a pirated copy with a null fingerprint for instance.

How many times do the colluders need to cheat? With probability p_i^c (resp. $(1-p_i)^c$), they all have bit '1' (resp. '0') at index i. Thus, there are on average $m_c(c) = m\int_t^{1-t}(p^c + (1-p)^c)f(p)dp$ such indices. The Provider asks for a bit verification with probability m_v/m_h. The probability of a successful attack for a collusion of size c is therefore $(1 - m_v/m_h)^{m_c(c)}$. Our numerical simulations (see figure 4 (a)) show that m_v shouldn't be more than 50 bits for typical code length and collusion size below a hundred. Thus, m_v is well below m_h.

Suppose now that the Provider is dishonest. The fact that the m lists $\mathcal{C}_i$, $\forall i \in [m]$ are public and not modifiable prevents the Provider from altering them for a specific Buyer in order to frame him/her afterwards. Moreover, it will raise the Judge's suspicion if the empirical distribution of the p_i is not close to the pdf f. Yet, biases can be introduced on the probabilities for the symbols of the colluders' fingerprint only if there is a coalition between them and the untrustworthy Provider. For instance, the Provider can choose a permutation such that by selecting the first item (resp. the last one) in the list $\mathcal{D}_{j,i}$ an accomplice colluder is sure to pick up a symbol '1' (resp. '0'). This ruins the tracing property of the code, but this does not allow the Provider to frame an innocent. First, it is guaranteed that $\mathbf{p}$ used in Eq. 1 is the one which generated the code. Second, the Provider and his accomplices colluders must ignore a significant part of the fingerprints of innocent Buyers. To this end, $m - m_h$ must also be in order of

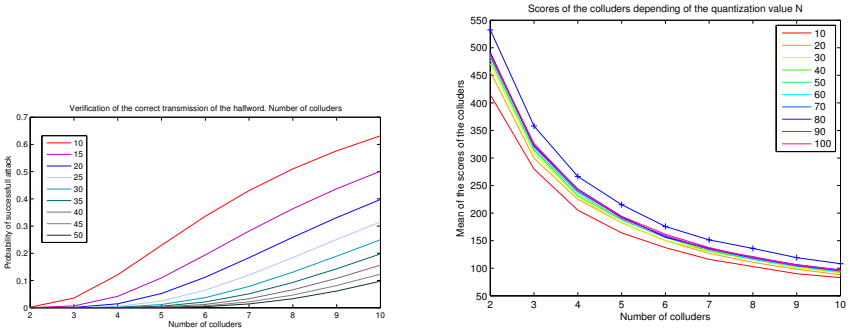

Fig. 4. (a) m_v goes from 10 to 50 by 5, $m = 3000$ and $m_h = 1500$. (b) N goes from 10 to 100 and $m = 1500$. The blue curve with crosses represents the unquantified Tardos code.

$O(c^2 \log n\epsilon^{-1})$. If this holds, the Judge is able to take a reliable decision while discarding the halfword part of the fingerprint. Consequently, $m \approx 2m_h$, our protocol has doubled the typical code length, which is still in $O(c^2 \log n\epsilon^{-1})$.

5.2 Efficiency

Parameters The parameters of the Tardos code are chosen according to the formulas linking length, number of colluders, and number of users. We have found out that the value m_v doesn't need to be more than 50, see Sec. 4. We consider the value N, the quantization parameter. In the figure 4 (b), we can see that up to a small value of N (around 20), there is no gain of efficiency. The red line shows that the results with the unquantized Tardos parameters remain better. Sans preciser l'attaque de la collusion, c'est un peu vaseux.

5.3 Complexity

A faire. Je ne vois pas l'utilite du tableau si le cost des OT n'est pas donne.

6 Conclusion

Tardos codes are currently the state-of-the-art in collusion-resistant fingerprinting. However, the previous asymmetric fingerprint protocols cannot be applied to this particular construction. There are mainly two difficulties. First, the Buyer has to generate his/her secret fingerprint but according to vector $\mathbf{p}$, which is kept secret by the Provider. Second, the secret $\mathbf{p}$ used in the accusation process must be the same as the one which generated the fingerprints. We have proposed the first asymmetric fingerprinting protocol dedicated to Tardos codes. The construction of the fingerprints and their embedding within pieces of Work do not need a trusted third party. Note, however, that during the accusation stage, a

Protocol	Cost	
	Commutative encryption	Any OT
generation of the fingerprint 1. Commitment phase Only one time for all users	$m \times N$ commitments	
For each user and each position		
2.	N encryptions (N exp)	
3.	1 encryption (1 exp)	
4.	1 decryption (1 exp)	
5.	1 decryption (1 exp)	1-out-of-N Oblivious transfer
6.	1 opening of commitment	
Generation of the halfword for each user		
1.	m commitments	
2.	m encryptions (m exp)	
3.	m_h encryption (m_h exp)	
4.	m_h decryption (m_h exp)	
5.	m_h decryption (m_h exp)	m_h-out-of-m Oblivious transfer
6.	m_h opening of commitment	

trusted third party is necessary like in any asymmetric fingerprinting scheme we are aware of. Further work is needed to determine if such a third-party can be eliminated. In particular, we anticipate that some form of secure multi-party computation can be applied.

References

1. D. Boneh and J.Shaw., "Collusion-secure fingerprinting for digital data," *IEEE Trans. Inform. Theory*, 1998.
2. G. Tardos, "Optimal probabilistic fingerprint codes," in *STOC 2003*. ACM, 2003, pp. 116–125.
3. B. Pfitzmann and M. Schunter, "Asymmetric fingerprinting," in *EUROCRYPT 96*. 1996, vol. 1070 of *LNCS*, pp. 84–95, Springer-Verlag.
4. B. Skoric, S. Katzenbeisser, and M. Celik, "Symmetric Tardos fingerprinting codes for arbitrary alphabet sizes," *Designs, Codes and Cryptography*, vol. 46, no. 2, pp. 137–166, 2008.
5. M. Rabin, "How to exchange secrets by oblivious transfer," Tech. Rep., Technical Report TR-81, Harvard Aiken Computation Laboratory, 1981, 1981.
6. M. Naor and B. Pinkas, "Oblivious transfer with adaptive queries," in *Advances in CryptologyCRYPTO99*. Springer, 1999, pp. 791–791.
7. C.K. Chu and W.G. Tzeng, "Efficient k-out-of-n oblivious transfer schemes with adaptive and non-adaptive queries," *Public Key Cryptography-PKC 2005*, pp. 172–183, 2005.
8. M. Green and S. Hohenberger, "Blind identity-based encryption and simulatable oblivious transfer," in *Proceedings of the Advances in Crypotology 13th international conference on Theory and application of cryptology and information security*. Springer-Verlag, 2007, pp. 265–282.

9. F. Bao, R.H. Deng, and P. Feng, "An efficient and practical Scheme for Privacy Protection in the E-Commerce of Digital Goods," in *ICISC 2000*. 2001, vol. 2015 of *LNCS*, pp. 162–170, Springer-Verlag.
10. J.G. Choi, J.H. Park, and K.R. Kwon, "Analysis of cot-based fingerprinting schemes: New approach to design practical and secure fingerprinting scheme," in *Information Hiding*. Springer, 2005, pp. 275–281.
11. Q.H. Wu, J.H. Zhang, and Y.M. Wang, "Practical t-out-n oblivious transfer and its applications," *Information and Communications Security*, pp. 226–237, 2003.
12. B. Zhang, H. Wu, D. Feng, and F. Bao, "Cryptanalysis of a knapsack based two-lock cryptosystem," in *Applied Cryptography and Network Security*. Springer, 2004, pp. 303–309.
13. H.F. Huang and C.C. Chang, "A new design for efficient t-out-n oblivious transfer scheme," 2005.
14. M. Naor and B. Pinkas, "Computationally secure oblivious transfer," *Journal of Cryptology*, vol. 18, no. 1, pp. 1–35, 2005.
15. J. Camenisch, G. Neven, and A. Shelat, "Simulatable adaptive oblivious transfer," *Advances in Cryptology-EUROCRYPT 2007*, pp. 573–590, 2007.
16. R. Canetti, "Universally composable security: A new paradigm for cryptographic protocols," in *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*. IEEE, 2002, pp. 136–145.
17. W. Ogata and K. Kurosawa, "Oblivious keyword search," *Journal of Complexity*, vol. 20, no. 2-3, pp. 356–371, 2004.
18. M. Naor and B. Pinkas, "Oblivious transfer and polynomial evaluation," in *Proceedings of the thirty-first annual ACM symposium on Theory of computing*. ACM, 1999, p. 254.
19. O. Goldreich, *Foundations of cryptography: Basic applications*, Cambridge Univ Pr, 2004.
20. H.C.A. van Tilborg, *Encyclopedia of cryptography and security*, Springer Verlag, 2005.
21. C. Fontaine and F. Galand, "A survey of homomorphic encryption for nonspecialists," *EURASIP Journal on Information Security*, vol. 2007, pp. 15, 2007.
22. A. Oprea and K. D. Bowers, "Authentic Time-Stamps for Archival Storage," in *ESORICS 2009*. 2009, vol. 5789 of *LNCS*, pp. 136–151, Springer-Verlag.
23. M. Kuribayashi, "On the Implementation of Spread Spectrum Fingerprinting in Asymmetric Cryptographic Protocol," *EURASIP Journal on Inf. Security*, 2010.
24. M. Deng, T. Bianchi, A. Piva, and B. Preneel, "An efficient Buyer-Seller watermarking protocol based on composite signal representation," in *ACM MM&Sec'09*, 2009, pp. 9–18.
25. T. Furon and L. Pérez-Freire, "Worst case attack against binary probabilistic traitor tracing codes," in *IEEE WIFS 2009*, 2009, pp. 46–50.