Tardos codes for real

Teddy Furon, Mathieu Desoubeaux

Tardos codes for real

Teddy Furon*, Mathieu Desoubeaux†

* *Inria Rennes, Campus de Beaulieu, Rennes, France*
† *LAMARK, Rennes, France*

***Abstract*—This paper deals with active fingerprinting a.k.a. traitor tracing where a collusion of dishonest users merges their individual versions of a content to yield a pirated copy. The Tardos codes are one of the most powerful tools to fight against such collusion process by identifying the colluders.**

Instead of studying as usual the necessary and sufficient code length in a theoretical setup, we adopt the point of view of the practitioner. We call this the *operational mode*, *i.e.* a practical setup where a Tardos code has already been deployed and a pirated copy has been found.

This new paradigm shows that the known bounds on the probability of accusing an innocent in the theoretical setup are way too pessimistic. Indeed the practitioner can resort to much tighter bounds because the problem is fundamentally much simpler under the operational mode.

In the end, we benchmark under the operational mode several single decoders recently proposed in the literature. We believe this is a fair comparison reflecting what matters in reality.

I. Introduction

The Tardos codes are the most popular fingerprinting codes, being the subject of a strong and living literature. From its invention in 2003 by G. Tardos, it has been deeply studied from the information theoretical and the statistical points of view, with many possible extensions. This paper sheds some light with a different angle: the operational mode.

In a nutshell, a typical theoretical paper about Tardos codes (like [16], [14], [15], [12], [10]) can be seen as mathematical function whose inputs are the following requirements:

- n the number of users,
- c the size of the collusion,
- ϵ_1 the probability of accusing a given innocent user,
- ϵ_2 the probability of missing a given guilty user.

The output is the necessary code length m to fulfill the above requirements: the theoretical paper proves that if m is bigger than $M(n, c, \epsilon_1, \epsilon_2)$, then *whatever* the attack strategy of the colluders, we will almost surely trace back some colluders while not accusing an innocent.

The main contribution of our paper is to stress that this setup is valid for a theoretical study, but it doesn't capture what matters at the decoder side in real life applications. Sect. II presents the shift of paradigm from the theoretical setup to what we call *the operational mode* and lists the main differences between the two points of view.

The value of utmost importance in the operational mode is the probability that an innocent user has a score higher than a given threshold. Sect. III reviews well-known upper bounds on and estimators of this probability. Some of them have been already used in classical Tardos codes theoretical literature. We here show how these bounds get more accurate in the operational mode. Another contribution is the introduction of other bounds dedicated to this new paradigm.

A third contribution is a comparison of several provably good single decoders recently proposed in the literature. It is somewhat difficult to compare them because their inventors claim optimality under different criteria. Our benchmark tuned for the operational mode provides a fair comparison in Sect. V.

WIFS'2014, December 3-5, 2014, Atlanta, USA.

Supported by french project Secular ANR-12-CORD-0014.

II. Tardos code: Theory vs. Practice

To make our point clear, we shall start with some classical notations for a binary Tardos code.

A. Notations

A is a random variable with expectation $\mathbb{E}(A)$ and variance $\mathbb{V}(A)$, a is a scalar, and $\mathbf{a}$ denotes vector $(a_1, \ldots, a_m)$. For any integer $m \geq 1$, $[m]$ denotes the set $\{1, \ldots, m\}$.

The code designer first randomly draws a vector $\mathbf{p} = (p_1, \ldots, p_m)$ whose components are random variables independently distributed with a probability density function $f_\tau : \mathcal{P}_\tau \to \mathbb{R}^+$. The support of function f_τ is $\mathcal{P}_\tau = (\tau, 1-\tau) \subset (0, 1)$. Vector $\mathbf{p}$ is often called the bias vector and it must be kept secret. The integer m is the length of the code. Any user j willing to receive a copy is associated to a binary codeword $\mathbf{x}_j = (x_{1,j}, \ldots, x_{m,j})$. The i-th bit $x_{i,j}$ is the result of a random draw according to Bernoulli law $\mathcal{B}(p_i)$ (*i.e.* $\mathbb{P}(X_{i,j} = x) = p_i^x (1-p_i)^{(1-x)}$). This bit is independent from the other symbols of $\mathbf{x}_j$ and of the other codewords. The code is denoted by $\mathcal{X} = \{\mathbf{x}_j\}_{j=1}^n$ and it must be kept secret in the sense that user j might only know $\mathbf{x}_j$.

A watermarking technique embeds the m-bit codeword $\mathbf{x}_j$ in the content and the results is sent to user j. A group of c dishonest users, so called collusion and denoted by $\mathcal{C}$, then merges their copies in order to forge a pirated copy with the hope that none of them can be traced back.

Once a pirated copy is found, the watermark decoder extracts the sequence $\mathbf{y} = (y_1, \ldots, y_m)$. A tracing algorithm has the three following inputs $(\mathbf{y}, \mathbf{p}, \mathcal{X})$. It aims at identifying

some guilty users, *i.e.* the members of the collusion. To do so, it computes a score s_j for any user j and it accuses those whose score is above a given threshold z.

B. Tardos code in theory

Most papers dealing with Tardos codes adopt an information theoretical or statistical point of view. They are based on a setup where the above mentioned variables are modeled as random variables: $\mathbf{P}, \mathbf{X}_j, \mathbf{Y}, S_j$. In the binary case, they often assume a statistical model $\boldsymbol{\theta}_c = (\theta_{0,c}, \ldots, \theta_{c,c})$ capturing the nature of the collusion attack [6]:

$$\theta_{s,c} = \mathbb{P}\left(Y_i = 1 | \sum_{j \in \mathcal{C}} X_{j,i} = s\right), \forall s \in \{0, \ldots, c\}. \quad (1)$$

In words, $\theta_{s,c}$ is the probability that we decode symbol '1' at indexes where the colluders have s times out of c symbol '1'.

Without loss of generality, we restrict our analysis to the case where the tracing algorithm is based on a single decoder. This means that the score of user j doesn't depend on the code $\mathcal{X}$ other than by his codeword $\mathbf{X}_j$. This implies that $S_j = G(\mathbf{X}_j, \mathbf{Y}, \mathbf{P})$ is a random variables with complex dependencies on $\mathbf{X}_j, \mathbf{Y}$, and $\mathbf{P}$.

Assuming at most c colluders among n users, a typical theoretical paper exhibits a density f_τ and a threshold z s.t.:

$$\mathbb{P}(S_{j \notin \mathcal{C}} > z) < \epsilon_1, \quad \text{[Proof of soundness]} \quad (2)$$
$$\mathbb{P}(S_{j \in \mathcal{C}} < z) < \epsilon_2, \quad \text{[Proof of completeness]} \quad (3)$$

whatever the collusion attack, provided that the code length is bigger than $m = M(n, c, \epsilon_1, \epsilon_2)$. The threshold z is 'universal' in the sense that for any occurrence taken by $\mathbf{P}$, $\mathbf{X}_j$, and $\mathbf{Y}$ complying with the theoretical model, (2) and (3) hold.

C. Tardos code in practice

In words, a theoretical paper takes the point of view of the code designer willing to deploy a fingerprinting solution. In this paper, the *operational mode* denotes the point of view of the decoder where a system *has already been deployed*.

In the operational mode, the setup of is very different: A content has been distributed to n users. The watermarking technique in use has a given bitrate $1/\ell$: it hides a binary symbol in a robust and imperceptible manner in every ℓ-samples of content (expressed in seconds, pixels, *etc.*). Given the size L of the content, the code length is thus $m = L/\ell$. Therefore, m is *not* an output, *i.e.* a parameter depending on other parameters, but an input in our scenario.

The decoder might not know the distribution f_τ or how suitable is this f_τ. It only knows $\mathbf{p}$ and $\mathcal{X}$. The decoder has a priori no clue on the collusion size c and the collusion strategy. It only observes its result $\mathbf{y}$. However, it must not accuse a given innocent user or with a probability at most ϵ_1. This means that it should automatically 'notice' and give up when the code length or the distribution f_τ is not suitable for reliably accusing colluders in the observed context. To summarize, the inputs of the problem in practice are $(n, m, \mathbf{p}, \mathbf{y}, \mathcal{X}, \epsilon_1)$.

In the operational mode, there is thus no randomness. However, since we don't know which codewords in $\mathcal{X}$ contributed to the forgery $\mathbf{y}$, we consider that the codeword of an innocent is a random variable $\mathbf{X}$ distributed according to the secret bias vector $\mathbf{p}$: $X_i \sim \mathcal{B}(p_i)$, $\forall i \in [m]$. The score of this innocent is random and equals $S = G(\mathbf{X}, \mathbf{y}, \mathbf{p})$. We drop the index j to insist on the fact that $\mathbf{X}$ and S are not related to a given user in particular, but are related to an innocent in general. Please, note that in the operational mode, there is only one source of randomness in the score computation.

This shift of paradigm deeply modifies the goal of the study. We no longer look for a 'universal' threshold z guaranteeing (2) and (3) for any realizations of $\mathbf{X}$, $\mathbf{P}$ and $\mathbf{Y}$, but a particular threshold that only holds for $\mathbf{y}$ and $\mathbf{p}$. We can do even better than this. Ultimately, we look for a function that maps the score s_j of user j to a probability of being innocent, depending on the observations $\mathbf{y}$ and $\mathbf{p}$. In Sect. IV, we propose the following mapping $\Pi(s_j) = \mathbb{P}(S > s_j | \mathbf{y}, \mathbf{p})$.

III. Bounds and Estimations

This section reviews some techniques for bounding or estimating the probability that the score of an innocent is bigger than a threshold. We focus on single decoders with linear score: $s_j = \sum_{i=1}^m u_i$, where $u_i = g(x_{j,i}, y_i, p_i)$. Function g is called the accusation score function. There is a vast literature about probabilities upper bounds for sums of independent random variables. The book [2] is dedicated to this subject. Most of them are meant for random variables with bounded support like (4) and (5) detailed hereafter.

A. Bounds for the theoretical setup

In the theoretical setup, $S = \sum_{i=1}^m U_i$ with $U_i = g(X_i, Y_i, P_i)$. Thanks to the code construction, U_i are mutually independent. Bounding the probabilities dealing with score has been the core of theoretical proofs in Tardos code literature. Yet, it is not easy to find tight bounds because U_i is a continuous random variable with a complex law due to its dependency on X_i, P_i, and Y_i. The Ph.D. thesis of A. Simone [13] is indeed devoted to the study of the distribution of U_i.

1) Tardos' inequality: In his seminal paper [16], Tardos uses this bound: If $\exists\, b > 0$ s.t. $U_i - \mathbb{E}(U_i) < b$, $\forall i \in [m]$, then

$$\mathbb{P}(S > z) = \begin{cases} e^{-\frac{(z-\mathbb{E}(S))^2}{2\sum_i \mathbb{V}(U_i)}} & \text{if } z < \frac{3.4\sum_i \mathbb{V}(U_i)}{b} + \mathbb{E}(S) \\ e^{-\frac{1.7(z-\mathbb{E}(S))}{b} + \frac{2.89\sum_i \mathbb{V}(U_i)}{b^2}} & \text{otherwise.} \end{cases} \quad (4)$$

2) Bernstein's inequality: Bernstein's inequality has been recently proposed as an alternative of (4) to simplify proofs [15]. If $\exists\, b > 0$ such that $|U_i| < b$, $\forall i \in [m]$, then

$$\mathbb{P}(S > z) \leq \exp\left(-\frac{(z - \mathbb{E}(S))^2/2}{\sum_i \mathbb{V}(U_i) + b(z - \mathbb{E}(S))/3}\right). \quad (5)$$

In the theoretical setup, these bounds are very relevant for accusation score function (14), because for an innocent user, $\mathbb{E}(S) = n\mathbb{E}(U_i) = 0$ and $\mathbb{V}(U_i) = 1$ whatever the collusion attack. Besides, since by construction $\tau < P_i < 1 - \tau$, we have $|U_i| < b = \sqrt{(1-\tau)/\tau}$. However, as soon as we use

a different accusation score function, all these nice properties vanish and the expectation and variance of U_i do depend on the collusion strategy, unknown at the decoder side. This explains the long lasting success of score function (14).

B. Bounds for the operational mode

In the operational mode, we now have $U_i = g(X_i, p_i, y_i)$ where (p_i, y_i) are known and $X_i \sim \mathcal{B}(p_i)$. R.v. U_i is now discrete taking two values, $a_i = g(0, p_i, y_i)$ and $b_i = g(1, p_i, y_i)$:

$$U_i = (b_i - a_i)X_i + a_i. \tag{6}$$

This simplifies a lot the bounds. For instance, U_i has for expectation and variance:

$$\mathbb{E}(U_i) = a_i + p_i(b_i - a_i), \tag{7}$$
$$\mathbb{V}(U_i) = p_i(1-p_i)(b_i - a_i)^2, \tag{8}$$

which in turn gives $\mathbb{E}(S) = \sum_{i=1}^m \mathbb{E}(U_i)$ and $\mathbb{V}(S) = \sum_{i=1}^m \mathbb{V}(U_i)$ thanks to the mutual independence. Setting $b = \max_i(\max(|a_i|, |b_i|))$, we can now apply the previous bounds even if the accusation score function is not given by (14) and without knowing the collusion attack. Another approach is to benefit from bounds dedicated to binary r.v. U_i as in (6).

1) Kearns and Saul's inequality [9, Th. 2]: For all $1 \leq i \leq m$, denote $d_i = b_i - a_i$. Then, for $z > \mathbb{E}(S)$:

$$\mathbb{P}(S > z) < \exp\left(-\frac{(z - \mathbb{E}(S))^2}{\sum_{i=1}^m \beta_i}\right) \tag{9}$$

$$\text{where } \beta_i = \begin{cases} 2d_i^2 & \text{if } p_i = 1/2 \\ d_i^2 \frac{1-2p_i}{\log(1-p_i) - \log(p_i)} & \text{otherwise.} \end{cases} \tag{10}$$

2) Cramer-Chernoff's inequality [2, Sect. 2.2]: The Cramer-Chernoff's bounding method has no closed form but requires a simple numerical optimization in the binary case. For $\lambda > 0$, Markov inequality yields:

$$\mathbb{P}(S > z) < \exp(-(\lambda z - \Psi_S(\lambda))) \tag{11}$$

where $\Psi_S(\lambda)$ denotes the logarithm of the moment generating function: $\Psi_S(\lambda) = \log(\mathbb{E}(e^{\lambda S}))$. Thanks to the mutual independence, $\Psi_S(\lambda) = \sum_{i=1}^m \Psi_{U_i}(\lambda)$. From (6), it comes

$$\Psi_{U_i}(\lambda) = -\lambda p_i(b_i - a_i) + \log\left(1 + p_i e^{\lambda(b_i - a_i)} - p\right). \tag{12}$$

We then use Matlab to find out $\lambda^\star > 0$ giving the tightest bound, *i.e.* maximizing the quantity $[\lambda z - \Psi_S(\lambda)]$.

C. Estimation for the operational mode

The above inequalities apply to scores if the decoder is single and linear. Yet, we also consider non linear single decoders (18) and (19) in our benchmark. Another approach is to estimate the probability $\mathbb{P}(S > z)$.

1) Monte Carlo: A naive method is a Monte-Carlo simulation. It consists in generating N new codewords $\{\tilde{\mathbf{x}}_j\}_{j=1}^N$. Sequence $\mathbf{y}$ was forged before, they are codewords of innocent 'users'. An estimation is $\hat{\mathbb{P}}(S > z) = N^{-1}|\{j | G(\tilde{\mathbf{x}}_j, \mathbf{p}, \mathbf{y}) > z\}|$. This is not efficient as $N = O(1/\mathbb{P}(S > z))$ which becomes intractable if $\mathbb{P}(S > z) < 10^{-9}$.

2) Rare event simulation: A faster algorithm was proposed in [3] which is based on rare event simulation. We don't describe it here and refer the reader to [3] for a complete description, [8] for the proof of its properties, and [5] for a Matlab implementation. Its key properties are that its estimation only consumes $O(-\log \mathbb{P}(S > z))$ scores computation and it provides a γ confidence interval $[P^{(-)}, P^{(+)}]$. In the sequel, we use the pessimistic estimation $\hat{\mathbb{P}}(S > z) = P^{(+)}$ for $\gamma = 0.95$. There is a probability of $(1-\gamma)/2 = 0.025$ that $P^{(+)}$ is indeed not an upper bound of $\mathbb{P}(S > z)$.

D. Examples

To illustrate the difference between the theoretical setup and the operational mode, we have generated a sequence $\mathbf{p}$ of length $m = 1,024$ with a density f_τ, and $c = 6$ codewords colluding via the interleaving attack ($\theta_{s,c} = s/c$). Figures 1 and 2 show the bounds and the estimations on $\mathbb{P}(S > z)$ for the accusation score functions (14) ($\tau = 3.3 * 10^{-4}$) and (15) ($\tau = 3.3 * 10^{-3}$) respectively.

As for the theoretical setup in fig. 1, the Tardos' bound is slightly tighter than the Bernstein inequality. The merit of this last bound only lies in the simplification of the proof of soundness [15]. These bounds are tighter in the operational mode than in the theoretical setup. However, the difference is tiny for Bernstein's inequality. The Cramer-Chernoff method offers the best bound at the cost of a numerical optimization.

In fig. 2, the inequalities (4) and (5) in the theoretical setup have two major drawbacks: i) since the accusation score function in use is not (14), they need to know the collusion attack to compute $\mathbb{E}(U_i)$ and $\mathbb{V}(U_i)$, ii) they are dramatically loose. In the operational mode, drawback i) no longer holds, but they are still very weak compared to the Kearn and Saul's inequality. Again the Cramer-Chernoff's method offers the best bound at the cost of a numerical optimization lasting ≈ 0.5s on our Matlab implementation.

The Monte Carlo simulation generates $N = 10^6$ codewords and scores, which takes $\approx$ 16s to estimate a probability down to 10^{-5}. The most reliable tool for estimating very low $\mathbb{P}(S > z)$ is the rare event simulation, which takes ≈ 11s for probabilities down to 10^{-11}. It is also the only approach that tackles scores which are not linear as described in Appendix B.

IV. A NEW DECODING RULE

The inequalities and estimation techniques under the operational setup offer new rules at the decoder side.

A. Adaptive threshold

The figures 1 and 2 were obtained under the theoretical setup and the operational mode, *i.e.* for a single run of a Tardos code simulation. They are thus specific for those particular realizations $\mathbf{p}$ and $\mathbf{y}$. From one run to another, the gap between the bounds (4) and (5) for the theoretical setup (valid for any $(\mathbf{p}, \mathbf{y})$) and the bounds or estimation in the operational mode (specific to $(\mathbf{p}, \mathbf{y})$) is more or less large. But the latter are always more precise and this has a dramatic impact of the decoding performances.

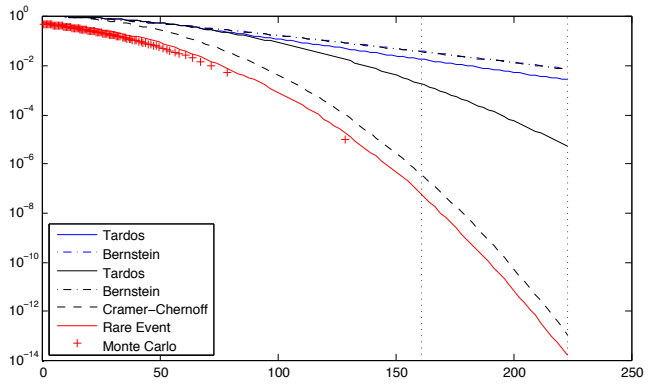

Fig. 1. Bounds, for the theoretical setup (blue) and the operational mode (black), and estimations (red) of $\mathbb{P}(S > z)$ for accusation score function (14). The minimum and the maximum of the $c = 6$ colluder scores are shown with dotted vertical lines.

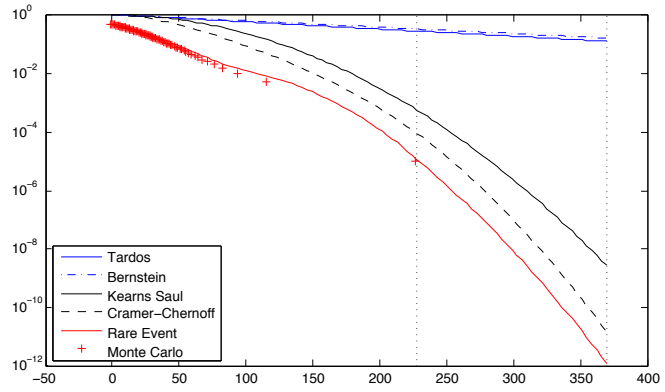

Fig. 2. Bounds, for the theoretical setup (blue) and the operational mode (black), and estimations (red) for accusation score function (15). The minimum and the maximum of the $c = 6$ colluder scores are shown with dotted vertical lines.

Suppose that we set $\epsilon_1 = 10^{-9}$ and $\epsilon_2 = 1/2$. The figures 1 and 2 show that m is not large enough to fulfill the requirements of the theoretical setup as stated in Sect. I: $\epsilon_1 = 10^{-9}$ implies a threshold z way too big to be on the figures according to the Tardos or Bernstein's inequalities (blue curves). At least for this realization of the code and pirated sequence $\mathbf{y}$, no colluders get caught because the maximum of the colluders scores is much smaller than z.

The conclusion is very different in the operational mode. The Cramer-Chernoff bound or the rare event estimator give a customized threshold $z(\mathbf{y}, \mathbf{p})$ such that $\mathbb{P}(S > z(\mathbf{y}, \mathbf{p})) < \epsilon_1$ which lies in between the minimum and the maximum of the colluder scores. Therefore, at least one colluder is caught in fig. 1 and 2. The notation $z(\mathbf{y}, \mathbf{p})$ stresses the fact that this threshold is only valid for these realizations of $(\mathbf{p}, \mathbf{y})$.

B. Probability of being innocent

Fig. 1 and 2 also offer a way to map a score s onto a probability of being innocent $\pi = \Pi(s)$. For user j, the bounds or estimations of Sect. III yield a probability $\pi_j = \Pi(s_j) = \mathbb{P}(S > s_j)$, which reads as the probability that the score of an innocent is higher than s_j. More precisely, they provide upper bounds mapping of the probability of being innocent with different tightness. In fig. 1, the probability that an innocent has a score as big as the maximum of the colluders' scores is around 10^{-2} according to Tardos' bound. Nobody would take the risk of accusing him if the probability of being wrong is only 10^{-2}. Yet, the rare event estimation yields a probability around 10^{-14}, which raises much suspicion.

This stresses the fact that we especially need tight bounds or accurate estimations for the biggest user scores. Figures 1 and 2 show that the bounds of the theoretical setup fail achieving this goal. As for the estimators, the computational inefficacy of the Monte-Carlo approach prevents accurate mapping for too big scores.

A tight mapping is very helpful to decide whether users with the biggest scores are to be accused because it is more meaningful than raw scores or their comparisons to a threshold. However, the probability π_j has to be related to the number of users n: The bigger n, the more likely at least one innocent user has a small π_j. Indeed if $c \ll n$, the probability that at least one innocent user has a mapping as low as π_j (equivalently a score as big as s_j) is $\eta_j = 1 - (1 - \pi_j)^n$. In fig. 1, the biggest colluders' score yields $\eta_j \approx n\pi_j = 10^{-8}$ if $n = 10^6$, which clears any doubt about his guiltiness.

V. Benchmark

This section compares the powers of the single decoders listed in appendix A.

A. Experimental setups

The code is constructed as described in Sect. II-A with the usual clipped arcsine distribution [16]:

$$f_\tau(p) = \frac{1}{2\arcsin(1-2\tau)\sqrt{p(1-p)}}, \forall p \in \mathcal{P}_\tau. \qquad (13)$$

We consider two setups with short and long codes. In setup $\mathcal{A}$, $m = 256$, $c = 3$, $\tau = 5.5 * 10^{-4}$, and $c_{\max} = 6$. In setup $\mathcal{B}$, $m = 1,024$, $c = 6$, $\tau = 3.3 * 10^{-4}$, and $c_{\max} = 10$. Parameter $c_{\max}$ is needed for decoders (16), (18), and (19). In a nutshell, these latter decoders bet that there is no use in looking for more than $c_{\max}$ colluders for such a code length.

These decoders are facing the following collusion attacks: Coin flip, all-1, all-0, interleaving, WCA, majority, and minority (see their definition in [6]). WCA is theoretically the worst case attack $\boldsymbol{\theta}_c^\star$ defined in [7] as the minimizer of the averaged mutual information per sample $\mathbb{E}_P(I(Y; X|P))$ for $P \sim f_\tau$.

B. Benchmark of decoders for a fixed f_τ

Our benchmark is composed of $N_r = 200$ runs. A run starts by generating $\mathbf{p}$ and c colluder codewords $\{\mathbf{x}_j\}_{j=1}^c$, which then forge $\mathbf{y}$ according to a given attack. The decoders considered in this benchmark compute the scores $\{s_j\}_{j=1}^c$ for this particular $\mathbf{y}$. Their minimum $s_{\min}$ and maximum $s_{\max}$ are then translated into probabilities $\Pi(s_{\min})$ and $\Pi(s_{\max})$ respectively thanks to the rare event simulator. We use the upper bound of its confidence interval as explained in Sect. III-C2.

Figures 3 and 4 show some statistics (median, 5% and 95% quantiles) about $\Pi(s_{\min})$ and $\Pi(s_{\max})$ over N_r simulation runs. Therefore, the best decoder is the one providing the smallest probabilities, which will trigger accusation.

For the first attacks (Coin flip, all-1, all-0, interleaving and WCA), decoders (16), (18), and (19) perform better. However, decoder (16) is weaker under the minority attack, whereas decoder (18) has troubles with the majority attack in Fig. 4 . If we assume that $n = 10^4$ with $\eta = 10^{-2}$ under setup $\mathcal{A}$, user j is accused if $\pi_j \lesssim 10^{-6}$. Only the decoders (18) and (19) succeed to accuse at least one colluder with a probability 1/2 (*i.e.* the median of $\Pi(s_{\max})$ is smaller than 10^{-6}) for all the tested attacks. If we assume that $n = 10^6$ with $\eta = 10^{-2}$ under setup $\mathcal{B}$, user j is accused if $\pi_j \lesssim 10^{-8}$. Only the decoder (19) succeeds to accuse at least one colluder with a probability 1/2, for all the tested attacks. Note however that the aggregated decoders (18) and (19) compute $c_{\max}$ times more scores than the other linear decoders (see Appendix B).

Finally, T. Laarhoven has discovered that his decoder (16) doesn't need a cutoff: It can handle f_τ with $\tau = 0$. Indeed, this also holds for decoders (18) and (19). However, we report that our benchmark with $\tau = 0$ shows a slight degradation of the performances for these three decoders. Here is the explanation: mutual information $I(X;Y|p)$ is weak for small p (or small $1-p$) and c not so large. Thus, generating too small p_i (or $1-p_i$) ruins the averaged mutual information $\mathbb{E}_P(I(Y;X|P))$.

VI. Conclusion

This paper motivates a shift of paradigm in the study of binary Tardos codes which occurs when taking the point of view at the decoder side. This new paradigm allows much more powerful bounds and estimations of the probability of accusing an innocent. It reveals that the bounds of the literature are very loose artificially increasing the necessary code length.

A benchmark under the operational mode compares some decoders recently proposed. It promotes the work of Desoubeaux *et al.* [4], which has so far received little attention.

Future works include the study of the Worst Case Forgery $\mathbf{y}$ for a given $\mathbf{p}$ and $\{\mathbf{x}_j\}_{j\in\mathcal{C}}$ (contrary to the Worst Case Attack $\boldsymbol{\theta}^\star_k$). The extension of our work to soft output watermarking decoder or to erasure / double symbols detection is straightforward. Yet, the extension to q-ary alphabet Tardos codes deserves deeper investigations.

VII. Acknowledgements

We would like to thank A. Guyader and B. Delyon for pointing out the Kearns and Saul's bound.

Appendix

We consider the following single decoders of two kinds: linear decoders and generalized linear decoders.

A. Linear decoders

A single decoder is linear if the score is a sum over m indexes: $s_j = \sum_{i=1}^m u_i$ with $u_i = g(x_{j,i}, y_i, p_i)$. The accusation score function is the name of $g : \{0,1\} \times \{0,1\} \times \mathcal{P}_\tau \to \mathbb{R}$.

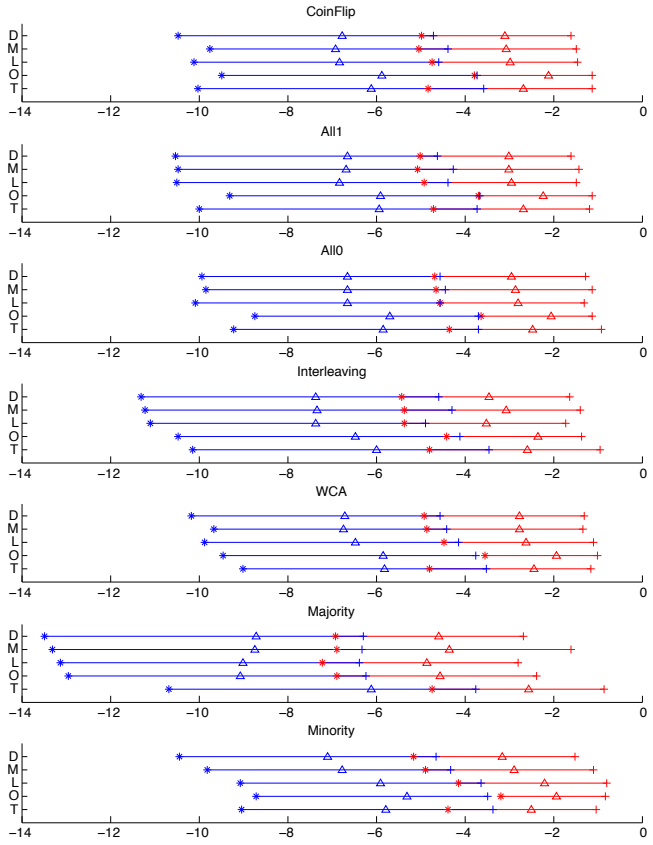

Fig. 3. Benchmark of single decoders under setup $\mathcal{A}$: (T) Tardos (14), (O) Oosterwijk (15), (L) Laarhoven (16), (M) Meerwald (18), (D) Desoubeaux (19). Statistics of $\log_{10}(\Pi(s_{\min}))$ (blue) and $\log_{10}(\Pi(s_{\max}))$ (red): Median ($\triangle$), 5% quantile ($*$), and 95% quantile ($+$).

1) Tardos-Skoric: Function g is given by:

$$g(x,y,p) = \begin{cases} \sqrt{(1-p)^{(2y-1)}p^{(1-2y)}} & \text{if } x = y, \\ -\sqrt{p^{(2y-1)}(1-p)^{(1-2y)}} & \text{if } x \neq y. \end{cases} \quad (14)$$

To bound its amplitude, we need to enforce $\mathcal{P}_\tau = [\tau, 1-\tau]$, where $0 < \tau < 1/2$ is the cutoff parameter. This function is optimal in the sense that, *whatever the collusion attack*, we have $\mathbb{E}(U_i) = 0$ and $\mathbb{V}(U_i) = 1$ allowing a straightforward application of bounds (4) and (5). Tardos initially proposed this function [16], later on generalized by Skoric *et al.* [14].

2) Oosterwijk et al. : Function g is given by [12, Eq. (43)]:

$$g(x,y,p) = \begin{cases} p^{-y}(1-p)^{y-1} - 1 & \text{if } x = y, \\ -1 & \text{if } x \neq y. \end{cases} \quad (15)$$

To bound its amplitude, we need to enforce a cutoff $\tau > 0$. This is optimal in the sense that it yields a saddle-point in the performance indicator $\mathbb{E}(\sum_{j\in\mathcal{C}} S_j)/\sqrt{\mathbb{V}(S_{j\notin\mathcal{C}})}$ [12, Th. 2], and that it is capacity-achieving [12, Prop. 17].

3) Laarhoven: Function g is given by [10, Eq. (2)]:

$$g(x,y,p) = \begin{cases} \log\left(1 + \frac{1}{c}\left(\frac{1-p}{p}\right)^{(2y-1)}\right) & \text{if } x = y, \\ \log(1-c^{-1}) & \text{if } x \neq y. \end{cases} \quad (16)$$

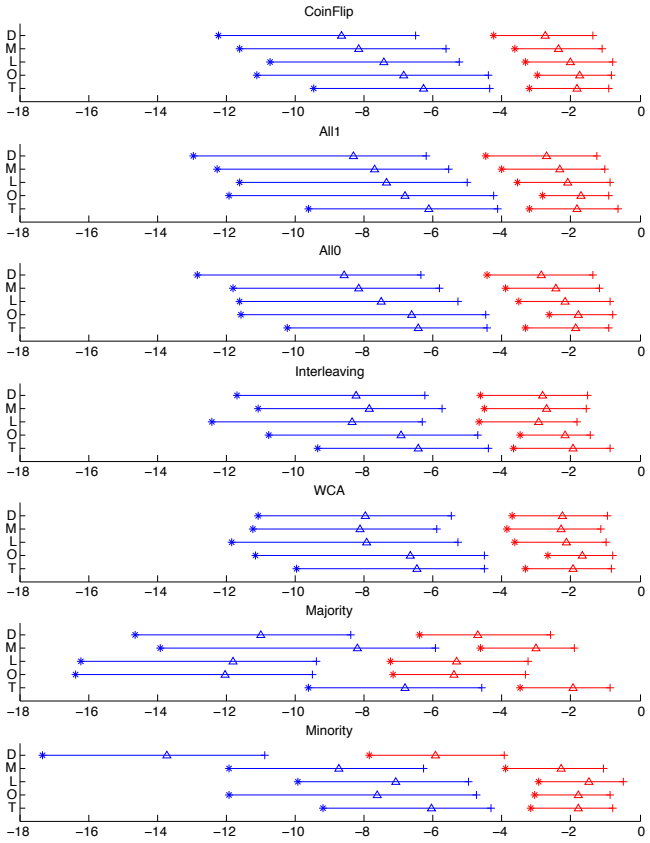

Fig. 4. Benchmark of single decoders under setup $\mathcal{B}$: (T) Tardos (14), (O) Oosterwijk (15), (L) Laarhoven (16), (M) Meerwald (18), (D) Desoubeaux (19). Statistics of $\log_{10}(\Pi(s_{\min}))$ (blue) and $\log_{10}(\Pi(s_{\max}))$ (red): Median ($\triangle$), 5% quantile ($*$), and 95% quantile ($+$).

This is optimal in the sense that it is capacity-achieving [10, Prop. 3]. Morever, there is no longer need of a cut-off parameter, *i.e.* $\mathcal{P} = (0, 1)$ [10, Th. 4]. Yet, the decoder needs to know c. In our simulation, the decoder bets that the collusion size is lower or equal to $c_{\max}$, *i.e.* c is replaced by $c_{\max}$ in (16).

4) MAP: This decoder is also known as ML or Neyman-Pearson decoder. The accusation function is given by:

$$g_{\boldsymbol{\theta}_c}(x, y, p) = \log\left(\frac{\mathbb{P}(Y = y|x, p, \boldsymbol{\theta}_c)}{(Y = y|p, \boldsymbol{\theta}_c)}\right) \tag{17}$$

The expression of the probabilities can be found in [7, Eq. (8-9)]. This is the best decoder to decide whether a given user is guilty from a statistical point of view. Yet, it needs the collusion size and attack model $\boldsymbol{\theta}_c$, which is not realistic in practice. Therefore, it is not part of our benchmark.

B. Generalized linear decoders

A decoder is a generalized linear decoder if the score is an aggregation of several linear scores: $s_j = A(s_j^{(1)}, \ldots, s_j^{(K)})$, with $K < +\infty$ and $A : \mathbb{R}^K \to \mathbb{R}$.

1) Meerwald & Furon: The decoder bets that there are at most $c_{\max}$ colluders. The aggregation function is the maximum operator:

$$s_j = \max(s_j^{(1)} \ldots, s_j^{(c_{\max})}), \tag{18}$$

where $s_j^{(k)}$ is the linear score based on the accusation function $g_{\boldsymbol{\theta}_k^\star}$ in (17) with $\boldsymbol{\theta}_k^\star$ being the worst case attack for a collusion of size k: $\boldsymbol{\theta}_k^\star = \arg\min_{\boldsymbol{\theta}_k} \mathbb{E}_P(I(Y; X|p, \boldsymbol{\theta}_k))$. This decoder proposed in [11] is capacity achieving thanks to theorem 1 in [1] on communication through compound channels.

2) Desoubeaux et al. : The decoder bets that there are at most $c_{\max}$ colluders. The aggregation is the following:

$$s_j = \log\left(\sum_{k=1}^{c_{\max}} k.\exp s_j^{(k)}\right), \tag{19}$$

where $s_j^{(k)}$ is the linear score based on the accusation function $g_{\boldsymbol{\theta}_k}$ given in (17) with $\boldsymbol{\theta}_k = (0, 1/2, \ldots, 1/2, 1)$, *i.e.* the 'Coin flip' attack. This decoder is optimal in the sense of a Bayesian decoder with the less informative prior on the collusion attack [4].

References

[1] E. Abbe and L. Zheng. Linear universal decoding for compound channels. *IEEE Trans. on Inf. Theory*, 56(12):5999–6013, december 2010.

[2] S. Boucheron, G. Lugosi, and P. Massart. *Concentration Inequalities: A Nonasymptotic Theory of Independence*. Oxford University Press, 2013.

[3] F. Cérou, T. Furon, and A. Guyader. Experimental assessment of the reliability for watermarking and fingerprinting schemes. *EURASIP Jounal on Information Security*, (ID 414962):12 pages, 2008.

[4] M. Desoubeaux, C. Herzet, W. Puech, and G. Le Guelvouit. Enhanced Blind Decoding of Tardos Codes with New Map-Based Functions. In *IEEE 15th International Workshop on Multimedia Signal Processing (MMSP)*, Pula, Italie, Oct. 2013.

[5] T. Furon. Rare event Matlab toolbox. http://people.rennes.inria.fr/Teddy.Furon/website/software.html, 2011.

[6] T. Furon, A. Guyader, and F. Cérou. On the design and optimisation of tardos probabilistic fingerprinting codes. In *Proc. of the 10th Information Hiding Workshop*, LNCS, Santa Barbara, Cal, USA, may 2008.

[7] T. Furon and L. Perez-Freire. Worst case attacks against binary probabilistic traitor tracing codes. Rejected by IEEE Trans. on Information Forensics and Security. Posted on arXiv, Aug. 2009.

[8] A. Guyader, N. Hengartner, and E. Matzner-Løber. Simulation and estimation of extreme quantiles and extreme probabilities. *Applied Mathematics & Optimization*, pages 1–26, apr 2011.

[9] M. Kearns and L. Saul. Large deviation methods for approximate probabilistic inference. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, UAI'98, pages 311–319, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[10] T. Laarhoven. Capacities and capacity-achieving decoders for various fingerprinting games. In *ACM Workshop on Information Hiding and Multimedia Security (IH&MMSec)*, June 2014.

[11] P. Meerwald and T. Furon. Towards practical joint decoding of binary tardos fingerprinting codes. *Information Forensics and Security, IEEE Transactions on*, PP(99):1, 2012.

[12] J.-J. Oosterwijk, B. Škorić, and J. Doumen. A capacity-achieving simple decoder for bias-based traitor tracing schemes. Cryptology ePrint Archive, Report 2013/389, 2013. http://eprint.iacr.org/.

[13] A. Simone. *Error probabilities in Tardos codes*. PhD thesis, T.U. Eindhoven, 2014.

[14] B. Škorić, S. Katzenbeisser, and M. Celik. Symmetric Tardos fingerprinting codes for arbitrary alphabet sizes. *Designs, Codes and Cryptography*, 46(2):137–166, February 2008.

[15] B. Škorić and J.-J. Oosterwijk. Binary and q-ary Tardos codes, revisited. *Designs, Codes and Cryptography*, pages 1–37, 2013.

[16] G. Tardos. Optimal probabilistic fingerprint codes. In *Proc. of the 35th annual ACM symposium on theory of computing*, pages 116–125, San Diego, CA, USA, 2003. ACM.