Towards practical joint decoding of binary Tardos fingerprinting codes

Peter Meerwald and Teddy Furon

Abstract—The class of joint decoder of fingerprinting codes is of utmost importance in theoretical papers to establish the concept of fingerprint capacity [1]–[4]. However, no implementation supporting a large user base is known to date. This article presents an iterative decoder which is the first attempt towards practical large-scale joint decoding. The discriminative feature of the scores benefits on one hand from the side-information of previously found users, and on the other hand, from recently introduced universal linear decoders for compound channels [5]. Neither the code construction nor the decoder make assumptions about the collusion size and strategy, provided it is a memoryless and fair attack. The extension to incorporate soft outputs from the watermarking layer is straightforward. An extensive experimental work benchmarks the very good performance and offers a clear comparison with previous state-of-the-art decoders.

Index Terms—Traitor tracing, Tardos codes, fingerprinting, compound channel.

I. INTRODUCTION

TRAITOR tracing or active fingerprinting has witnessed a flurry of research efforts since the invention of the now well-celebrated Tardos codes [6]. The codes of G. Tardos are order-optimal in the sense that the code length \( m \) necessary to fulfill the following requirements (\( n \) users, \( c \) colluders, probability of accusing at least one innocent below \( P_{fp} \)) has the minimum scaling in \( \Omega(c^2 \log n P_{fp}^{-1}) \).

A first group of articles analyses such probabilistic fingerprinting codes from the viewpoint of information theory. They define the worst case attack a collusion of size \( c \) can produce, and also the best defense. The main achievement is a saddle point theorem in the game between the colluders and the code designer which establishes the concept of fingerprinting capacity \( C(c) \) [1]–[4]. Roughly speaking, for a collusion of maximum size \( c \), the maximum number of users exponentially grows with \( m \) with an exponent equal to \( C(c) \) to guarantee vanishing probabilities of error asymptotically as the code length increases.

Our point of view is much more practical and signal processing oriented. In traitor tracing applications distinct codewords of \( m \) bits have been hidden in distributed copies with an appropriate watermarking technique. It implies that we have no choice on \( m \) as it depends on the content size and the watermarking embedding rate. Each embedded codeword links a copy of the content to a particular user. The total number of users may not be known in advance like, for instance, in a Video-on-Demand application where clients buy content sequentially. However, at the time a pirated version is discovered, we know that the content has been distributed to \( n \) users so far. Our goal is to identify some colluders under the strict requirement that the probability of accusing innocents is below \( P_{fp} \). It is clear that we are not in an asymptotic setup since \( m \) and \( n \) are fixed. The encoder and the decoder are not informed of the collusion size and its attack, therefore there is no clue whether the actual rate \( R = m^{-1} \log_2 n \) is indeed below capacity \( C(c) \). After reviewing the construction of Tardos codes and the collusion attack, Sec. II summarizes important elements of information theory as guidelines for the design of our decoder. It motivates the use of joint decoding, which computes a score per subset of users, as opposed to a single decoder computing a score per user.

A second group of related research works deals with decoding algorithms. As far as a practical implementation of joint decoding is concerned, the literature is very scarce. E. Amiri proposes a pair decoder tractable on a short list of suspects [7, Sec. 5.3]. The idea is proven to be theoretically well grounded against two colluders. However, the sorting is in terms of Hamming distance from the pirated sequence, which seems to be quite an ad hoc choice; no experimental work is conducted. K. Nuida also proposes a provably secure joint decoder against three colluders whose runtime is longer than one hour for a very small setup (\( n = 1000, m = 180 \)) [8].

For a single as well as a joint decoder, a primary challenge is to compute scores that are as discriminative as possible. The earliest decoders proposed in the literature are single decoders not adapting the score computation to the collusion strategy [6], [9]. They rely on an invariance property where, whatever the collusion process under the marking assumption, the statistics of scores of the innocents and the colluders are almost fixed and sufficiently discriminative if the code is long enough. Being inspired by a recent article on compound channel theory [5], we first propose a generalized linear single decoder which is more discriminative but at the cost of higher complexity. A second approach in the literature aims at first identifying the collusion process, and then at computing more discriminative scores for this specific attack [10]. However, the identifiability of the attack is a crucial issue when the number of colluders is not known. Again, the application of the concepts from [5] allows us to get inferences about the attack sufficient for deriving highly discriminative scores while avoiding a complete identification of the attack channel. Sec. III sums up these two families of single decoders and the
A further difficulty in traitor tracing schemes is the thresholding of the scores to reliably accuse users who are part of the collusion. The value of the threshold is easily set when the statistics of the scores are known, which is the case when the above-mentioned invariance property holds. However, for a general scoring function, these statistics depend on the collusion process which is not known. Section III-C presents a simple idea: there are plenty of codewords which have not been distributed to users. Therefore, it is possible to use them as instantiations of the codeword of an innocent. We propose to estimate the threshold yielding the required probability of false alarm with a rare event estimator.

Section IV focuses on the architecture of our joint decoder based on three primitives: channel inference, score computation, and thresholding. Its iterative nature stems from two ideas. First, the codeword of a newly accused user is integrated as a side information for the next iterations and enables more discriminative score computation. This idea was already implemented for fingerprinting codes based on error correcting codes [11]. We present a way to implement it for Tardos codes by conditioning the probabilities used in the score function. The second idea is joint decoding based on the channel inference.

A last difficulty is to have a fast implementation of the accusation algorithm when facing a large-scale user set. A main advantage of some fingerprinting schemes based on error-correcting codes is to offer an accusation procedure with a runtime polynomial in error-correcting codes [11], [12]. In comparison, the well-known Tardos-Škorić single decoder is based on an exhaustive search which has complexity $\Omega(mn)$ [6], [9]. Since in theory $n$ can asymptotically be in the order of $2^{nR}$, decoding of Tardos codes might be intractable. Again, we do not consider such a theoretical setup, but we pay attention to maintain an affordable decoding complexity for orders of magnitude met in practical applications. Compared to prior art of joint decoding [7], [8], our algorithm considers user subsets of bigger size, manages large scale setups, and is faster. Its iterative nature maintains a tractable complexity because users that are unlikely to be guilty are pruned out at each step.

Section V presents our experimental investigations. The first part relies on the marking assumption and compares code lengths with [13]. This reference reaches very small lengths thanks to a particular choice of Gauss-Legendre distribution, but assuming the collusion size is known at the Tardos code construction. It is interesting to see that our decoder obtains competitive lengths while keeping the original code construction. The second part uses the soft outputs of a watermarking decoder as tested in [14] with the Tardos code and, with error correction code (ECC) based fingerprinting codes, in [15], [16]. The number of users ranges from $10^4$ to more than $10^7$. This latter impressive setup comes from [16] where the authors manage such a large number of users thanks to list decoding of ECC based fingerprinting. As far as we know, our paper presents for the first time experimental results on such a large scale for the Tardos code. Soft watermark decoding achieves tangible performance enhancements contrary to the conclusions drawn in [14]. Overall, the comparisons to related works with their exact setup show the benefits of our decoder: better decoding performance with a controlled probability of false alarm and an acceptable runtime.

II. TARDOS CODE AND THE COLLUSION MODEL

We briefly review the construction and some well-known facts about Tardos codes.

A. Construction

The binary code is composed of $n$ codewords of $m$ bits. The codeword $x_i = (x_1(x), \ldots, x_i(m))^T$ identifying user $i \in \mathcal{U} = [n]$, where $[n] := \{1, \ldots, n\}$, is composed of $m$ binary symbols independently drawn at the code construction s.t. $P(x_i(t) = 1) = p_i$, $\forall i \in [n]$. At initialization, the auxiliary variables $\{p_i\}_{i=1}^m$ are independent and identically drawn according to distribution $f(p) : [0, 1] \to \mathbb{R}^+$. This distribution is a parameter which is public. G. Tardos originally proposed in [6] $f_T(p) \propto 1/\sqrt{p(1-p)}$ for $p \in [t, 1-t]$ where $t \ll 1$ is the cutoff parameter. Both the code $\Xi = [x_1, \ldots, x_n]$ and the auxiliary sequence $p = (p_1, \ldots, p_m)^T$ must be kept as secret parameters.

B. Collusion attack over code symbols

The collusion attack or collusion channel describes the way the $c$ colluders $\mathcal{C} = \{j_1, \ldots, j_c\}$ merge their binary codewords $x_{j_1}, \ldots, x_{j_c}$ to forge the binary pirated sequence $y$. We restrict our attention to a memoryless multiple access channel, which is fair in the sense that all colluders participate equally in the forgery. This assumption is widely used, and justified theoretically [1, Sec. 3.2] (in terms of capacity, i.e. asymptotically with the code length) when the secret key is only shared between encoder and decoder: The colluders know neither the codeword of any other user, nor the distribution of the codewords. This is the case for a Tardos code because $p$ is secret. Moreover, identifying all the colluders is hopeless (cf. detect-all scenario, see Sec. II-D) if the attack is not fair because some colluders might be almost idle [1, Lemma 3.2].

This leads to a $2 \times (c+1)$ probability transition matrix $[P(Y|\theta)]$, where $\Phi = \sum_{\ell \in \mathcal{C}} X_\ell$ is the random variable counting the number of ‘1’ symbols the colluders received out of $c$ symbols. A common parameter of the collusion attack on binary codes is denoted by the vector $\theta = (\theta_e(0), \ldots, \theta_e(c))^T$ with $\theta_c(\varphi) = P(Y = 1|\Phi = \varphi)$. The usual working assumption, so-called marking assumption [17], imposes that $\theta_e(0) = 1 - \theta_e(c) = 0$. The set of collusion attacks that $c$ colluders can lead under the marking assumption is denoted by $\Theta_c$:

$$\Theta_c = \{\theta \in [0, 1]^{c+1}, \theta(0) = 1 - \theta(c) = 0\}. \tag{1}$$

Examples of attacks following this model are given, for instance, in [10]. The remainder of the article assumes this collusion model, except for the simulations on real samples over an additive white Gaussian noise (AWGN) channel which are based on the extension presented below.
C. Collusion attack over real samples

The marking assumption is an unrealistic restriction for traitor tracing with multimedia content as the colluders are not limited to the copy-and-paste strategy for each symbol as described in the previous section. They can merge the samples of their content versions (audio samples, pixels, DCT coefficients, etc.) in addition to traditional attempts to compromise the watermark. This may result in erroneously decoded symbols or erasures from the watermarking layer. Relaxing the marking assumption leads to two approaches. In the combined digit model assumed in [18], [19], the watermark decoder is indeed composed of multiple binary detectors, one per symbol of the alphabets: for the binary alphabet, both symbols may be detected in case of a merge. In [14], [15], the watermark decoder has a single but scalar output $y'$. In brief, this soft decision is clearly negative (positive) if symbol ‘0’ (resp. ‘1’) is detected, and around 0 in case of a merge. This section extends the model of collusion to this latter approach, replacing the probability transition 2 $\times$ $(c+1)$ matrix $[P[Y|\theta]]$ by $c+1$ probability density functions $\{\theta_c(y'|\varphi)\}_{\varphi=0}^{\infty}$.

It is challenging if not impossible to exhibit a model encompassing all the merging attacks while being relevant for a majority of watermarking techniques. Our approach is pragmatic. The sequence $y' \in \mathbb{R}^m$ is extracted from the pirated copy such that $y'(i) = 2y(i) - 1$ if the signal is perfectly watermarked with binary symbol $y(i)$. To reflect the merging attack, the colluders forge values $z(i) \in [-1, 1]$ and add noise: $y'(i) = z(i) + n(i)$ with $n(i) \sim N(0, \sigma_2^2)$. This would be the case, as sketched in the left part of Fig. 3, for a spread spectrum watermarking where a symbol is embedded per block of content with an antipodal (a.k.a. BPSK, Binary Phase Shift Keying) modulation of a secret carrier [14], [16].

The colluders have two strategies to agree on $z$. In the first strategy, they collude according to the marking assumption (i.e. they copy-and-paste one of their blocks of samples) and add noise: $z \in \{-1, 1\}^m$ and the probability that $z = 1$ is given by the components of $\theta_c$. This gives the following pdf:

$$
\theta_c(y'|\varphi) = \left(\theta_c(\varphi)e^{-\frac{(y'-1)^2}{2\sigma_2^2}} + (1 - \theta_c(\varphi))e^{-\frac{(y'+1)^2}{2\sigma_2^2}}\right) / \sqrt{2\pi\sigma_2^2},
$$

(2)

Except for $\varphi \in \{0, 1\}$, the pdfs have a priori two modes (hence the superscript $I/II$). This model is parameterized by $(\theta_c, \sigma_2^2)$. In the second strategy, the colluders select $z(i) = \mu(\varphi(i)) \in [-1, 1]$ s.t. the pdf is as follows:

$$
\theta_c(y'|\varphi) = e^{-\frac{(y'-\mu(\varphi))^2}{2\sigma_2^2}} / \sqrt{2\pi\sigma_2^2}.
$$

(3)

An equivalent of the marking assumption would impose that $\mu(0) = -1$ and $\mu(1) = 1$. The pdfs have a unique mode (hence the superscript $I$). This model is parameterized by $(\mu, \sigma_2^2)$. Fig. 1 gives some examples of such pdfs.

We use these extended models for collusion inference in Sec. V-B to show how our algorithm can handle the soft outputs of a watermarking decoder.

D. Accusation

Denote $A \subset \mathcal{U}$ the set of users accused by the decoder. The probability of false positive is defined by $P_{fp} = \mathbb{P}(A \subset \mathcal{C})$. In practice, a major requirement is to control this feature so that it is lower than a given significance level.

In a detect-one scenario, $A$ is either a singleton, or the empty set. A good decoder has a low probability of false negative defined by $P_{fn} = \mathbb{P}(A = \emptyset)$. In a detect-many scenario, several users are accused, and a possible figure of merit is the number of caught colluders: $|A \cap \mathcal{C}|$. In the literature, there exists a third scenario, so-called detect-all, where a false negative happens if at least one colluder is missed. This article only considers the first two scenarios.

E. Guidelines from information theory

This article does not pretend to make any new theoretical contribution, but presents some recent elements to stress guidelines when designing our practical decoder.

A single decoder computes a score per user. It accuses users whose score is above a threshold (detect-many scenario) or the user with the biggest score above the threshold (detect-one scenario). Under both scenarios and provided that the collusion is fair, the performance of such decoders is theoretically bounded by the achievable rate $R_S(f, \theta_c) = I(X; Y|P, \theta_c) = \mathbb{E}_{P, f}[I(X; Y|P, \theta_c)]$. A fundamental result is that, for a given collusion size $c$, there exists an equilibrium $(\hat{f}, c, \hat{\theta}_c, \hat{s})$ to the max-min game between the colluders (who select $\theta$) and the code designer (who selects $f$) as defined by $\max_{f} \min_{\theta \in \Theta} R_S(f, \theta)$ in [2, Th. 4].

A joint decoder computes a score per subset of $\ell \leq c$ users and accuses the users belonging to subsets whose score is above a threshold or only the most likely guilty amongst these users. Under both scenarios and provided that the collusion is fair, the performance of such decoders is theoretically bounded by the achievable rate $R_J(f, \theta_c) = \ell^{-1}I(\hat{f}; Y|P, \theta_c) = \ell^{-1}\mathbb{E}_{P, f}[I(\hat{f}; Y|P, \theta_c)]$. The random variable $\Phi$ denotes the sum of the subset user symbols. Moreover, for a given collusion size $c$, there also exists an equilibrium $(\hat{f}, \theta_c, \hat{s})$ to the max-min game $\max_{f} \min_{\theta \in \Theta} R_J(f, \theta)$ [2, Th. 4].
Yet, the code designer needs to bet on a collusion size $c'$ in order to use the optimal distribution $f_{c',S}$ (or $f_{c',J}$ if the decoder is joint). Asymptotically, as $c \to +\infty$, both $\min_{\theta} R_{S}(f_{T}, \theta)$ and $\min_{\theta} R_{S}(f_{T}, \theta)$ quickly approach the equilibrium value of the respective max-min game [2, Fig. 2]. Y.-W. Huang and P. Moulin proved $f_{c',J}$ converges to $f_{T}$, the distribution originally proposed by G. Tardos [2, Cor. 7].

Despite the division by $\ell$ in the expression of $R_{J}(f, \theta)$, it appears that $R_{S}(f, \theta) \leq R_{J}(f, \theta), \forall \theta$ [1, Eq. (3.4)]. This tells us that a joint decoder is theoretically more powerful than a single decoder. However, a joint decoder needs to compute $\Omega(n^{\ell})$ scores since there are $\binom{n}{\ell}$ subsets of size $\ell$. This complexity is absolutely intractable for large-scale applications even for a small $\ell$. This explains why, so far, joint decoders were only considered theoretically to derive fingerprinting capacity.

Our idea is that there is no need to consider all these subsets since a vast majority is only composed of innocent users. Our decoder iteratively prunes out users deemed as innocents and considers the subsets over the small set of remaining suspects.

This iterative strategy results in a decoder which is a mix of single and joint decoding. Unfortunately, it prevents us from taking advantage of the game theory theorems mentioned above. We cannot find the optimal distribution $f$ and the worst collusion attack against our decoder. Nevertheless, our decoder works with any distribution $f$ under the conditions stated in Sec. III. For all these reasons, the experiments of Sec. V are done with the most common Tardos distribution $f_{T}$.

M. Fernandez and M. Soriano proposed an iterative accumulation process of an error correcting code based fingerprinting scheme [11]. Each iteration takes advantage of the codewords of colluders already identified in the previous iterations. This idea is possible with Tardos fingerprinting code. This is justified by the fact that the side information $\Delta$, defined as the random variable sum of the already identified colluder symbols, increases the mutual information $I(\Phi; Y|P, \theta_{c}) \leq I(\Phi; Y|P, \theta_{c}, \Delta)$. Indeed, side information helps more than joint decoding as proved by [1, Eq. (3.3)].

The above guidelines can be summarized as follows: use the continuous Tardos distribution $f_{T}$ for code construction, integrate the codewords of already identified colluders as side information, and finally use a joint decoder on a short list of suspects.

### III. A SINGLE DECODER BASED ON COMPOUND CHANNEL THEORY AND RARE EVENT ANALYSIS

This section first reviews some single decoders and presents new decoders based on compound channel theory and rare event analysis. The first difficulty is to compute a score per user such that the colluders are statistically well separated from the innocents scores. The second difficulty is to set a practical threshold such that the probability of false positive is under control.

Detection theory tells us that the score given by the Log-Likelihood Ratio (LLR):

$$s_{j} = \sum_{i=1}^{\min_{\theta} R_{S}(f_{T}, \theta)} \frac{\log \frac{P(g(i)|x_{j}(i), \theta_{c})}{P(g(i)|\theta_{c})}}{\max_{\theta} R_{S}(f_{T}, \theta)},$$

is optimally discriminative in the Neyman-Pearson sense to decide the guiltiness of user $j$. For the sake of a lighter expression, we omit the dependence of the involved probabilities on $p(i)$ (see their computation in Sub. IV-B). Yet, the LLR needs the knowledge of the true collusion attack $\theta_{c}$ which prevents the use of this optimal single decoder in practical settings. Some papers proposed a so-called ‘Learn and Match’ strategy using the LLR score tuned on an estimation $\widehat{\theta}$ of the attack channel [10]. Unfortunately, a lack of identifiability obstructs a direct estimation from $(y, p)$ (see Sec. III-B). Indeed, the estimation is sound only if $c$ is known, and if the number of different values taken by $p$ is bigger than $1$ or equal to $c - 1$. This is because $P(Y = 1|\theta, p)$ is a polynomial in $p$ of degree at most $c$ (see (16) with $u = 0$ and $v = 0$) going from point $0$ to $(1, 1)$, and we need $c - 1$ more points to uniquely identify this polynomial. To overcome this lack of information about $c$, an Expectation-Maximization (E-M.) approach has been proposed in [10]. Yet, it is not satisfactory since it does not scale well with the number of users. Moreover, the setting of the threshold was not addressed.

On the other hand, there are decoders that do not adapt their score computation to the collusion. This is the case of the score computation originally proposed by G. Tardos [6], and later on improved by B. Škorić et al. [9]. It relies on an invariance property: the statistics of the scores, up to the second order, do not depend on the collusion attack channel $\theta$, but only on the collusion size $c$ [20]. Thanks to this invariance w.r.t. the collusion attack, there exists a threshold $\tau$ guaranteeing a probability of false positive below $P_{fp}$ while keeping the false negative away from 1 provided that the code is long enough, i.e. $m = \Omega(c^{3} \log nP_{fp}^{-1})$. However, there is a price to pay: the scores are clearly less discriminative than the LLR.

Some theoretical papers [21, Sec. V] [1, Sec. 5.2] promote another criterion, so-called ‘universality’, for the design of decoders. The performance (usually evaluated as the achievable rate or the error exponent) when facing a collusion channel $\theta_{c}$ should not be lower than the performance against the worst attack $\theta_{c}^{*}$. In a sense, it sends a warning to the ‘Learn and Match’ strategy. Suppose that $\theta_{c} \neq \theta_{c}^{*}$ and that, for some reasons, the estimation of the collusion attack is of poor quality. In any case, a mismatch between $\theta$ and $\theta_{c}$, should not ruin the performance of the decoder to the point it is even lower than what is achievable under the worst attack $\theta_{c}^{*}$. The above cited references [1], [21] recommend the single universal decoder based on the empirical mutual information $I(x; y|p)$ (or empirical equivocation for joint decoder). The setting of the threshold depends on the desired error exponent of the false positive rate. Therefore, it is valid only asymptotically.

To summarize, there have been two approaches: adaptation or non-adaptation to the collusion process. The first class is not very well grounded since the identifiability of the collusion attack is an issue and the impact of a mismatch has not been studied. The second approach is more reliable, but with a loss of discrimination power compared to the optimal LLR. The

1This is the case in this article since we opt for the continuous Tardos distribution $f_{T}$. 
next sections present two new decoders belonging to both approaches based on the compound channel theory.

A. Some elements on compound channels

Recently, in the setup of digital communication through compound channels, E. Abbe and L. Zheng [5] proposed universal decoders which are linear, i.e. in essence very simple. This section summarizes this theory and the next one proposes two applications for Tardos single decoders.

A compound channel $\Psi$ is a set of channels, here discrete memoryless channels $X \in \mathcal{X} \to Y \in \mathcal{Y}$ defined by their probability transition matrix $W_\psi = [P(Y|X,\psi)]$ parameterized by $\psi \in \Psi$. The encoder shares a code book $\Xi = \{X_j\}_{j=1}^n \in \mathcal{X}^n \times \mathcal{Y}$ with the decoder. Its construction is assumed to be a random code realization from a provably good mass distribution $P_X$. After receiving a channel output $y \in \mathcal{Y}^n$, the decoder computes a score per codeword $x_j$, $j \in [n]$, and yields the message associated with the codeword with the biggest score. The decoder is linear if the score has the following structure:

$$s_j = \sum_{i=1}^m d(x_j(i), y(i)), \quad (5)$$

d with $d(\cdot, \cdot) : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$. For instance, score (4), so-called MAP decoder in digital communications [5], is linear with $d(x, y) = \log(P(y|x,\psi)/P(y|\psi))$. However, in the compound channel setup, the decoder does not know through which channel the codeword has been transmitted, and therefore it cannot rely on the MAP.

We are especially interested in two results. First, if $\Psi$ is one-sided with respect to the input distribution (see Def. 1 below), then the MAP decoder tuned on the worst channel $W_{\psi_k}$, is a linear universal decoder [5, Lemma 5]. If $\Psi = \bigcup_{k=1}^K \Psi_k$ with $K$ finite and $\Psi_k$ one-sided w.r.t. the input distribution $\forall k \in [K]$, then the following generalized linear decoder is universal [5, Th. 1] where the score of a codeword is the maximum of the $K$ MAP scores tuned on the worst channel $W_{\psi_k}$ of each $\Psi_k$:

$$s_j = \max_{k \in [K]} \sum_{i=1}^m \log P(y(i)|x_j(i), \psi_k^*).$$

Definition 1 (One-sided set, Def. 3 of [5]): A set $\Psi$ is one-sided with respect to an input distribution $P_X$ if

- the following minimizer is unique:

$$\psi^* = \arg \min_{\psi \in \mathcal{C}(\Psi)} I(P_X, \psi), \quad (7)$$

with $I(P_X, \psi)$ the mutual information $I(X;Y)$ with $(X, Y) \sim P_X \circ W_{\psi}$ (where $P \circ W$ denotes the joint distribution with $P$ the distribution of $X$ and $W$ the conditional distribution), and $\mathcal{C}(\Psi)$ the closure of $\Psi$,

- and $\forall \psi \in \Psi$,

$$D(P_X \circ W_{\psi} || P_X \times P_Y, \psi^*) \geq D(P_X \circ W_{\psi} || P_X \circ W_{\psi^*}) + D(P_X \circ W_{\psi^*} || P_X \times P_Y). \quad (8)$$

with $D(\cdot||\cdot)$ the Kullback-Leibler distance, $P_Y$ the marginal of $Y$ induced by $P_X \circ W_{\psi}$, and $P_X \times P_Y$ the product of the marginals.

B. Application to single Tardos decoders

Contrary to the code construction phase, it is less critical at the decoding side to presume that the real collusion size $c$ is less or equal to a given parameter $c_{\text{max}}$. This parameter can be set to the largest number of colluders the fingerprinting code can handle with a reasonable error probability knowing $(m, n)$. Another argument is that this assumption is not definitive. If the decoding fails because the assumption does not hold true, nothing prevents us to re-launch decoding with a bigger $c_{\text{max}}$. Let us assume $c \leq c_{\text{max}}$ in the sequel.

1) Non-adaption to the collusion process: A first guideline inspired from the work [5] is straightforward: The collusion channel belongs to the set $\bigcup_{k=1}^{c_{\text{max}}} \Theta_k$ as defined in (1), and thanks to [5, Lemma 4] each convex set $\Theta_k$ is one-sided w.r.t. any distribution $f$. According to [5, Th. 1], the detector should then be a generalized linear decoder:

$$s_j = \max_{k \in [2, \ldots, c_{\text{max}}]} \sum_{i=1}^m \log \frac{P(y(i)|x_j(i), \theta_{k,f}^*)}{P(y(i)|\theta_{k,f})},$$

where $\theta_{k,f}^* = \arg \min_{\theta \in \mathcal{E}(\Theta_k)} R_S(f, \theta), \forall k \in [2, \ldots, c_{\text{max}}]$. This decoder does not adapt its score computation to the collusion attack.

2) Adaption to the collusion process: The second idea inspired from the work [5] is more involved as the lack of identifiability turns to our advantage. The true collusion channel $\theta_c$ has generated data $y$ distributed as $P(y|p, \theta_c)$. Let us define the class $\mathcal{E}(\Theta_c) = \{\theta | P(y|p, \theta) = P(y|p, \theta_c), \forall (y, p) \in \{0, 1\} \times \{0, 1\}\}$. From [22, Prop. 3], we know that $\mathcal{E}(\Theta_c)$ is not restricted to the singleton $\{\theta_c\}$ because for any $c' > c$ there exists one $\theta_{c'} \in \mathcal{E}(\Theta_c)$. This is true especially for $c' = c_{\text{max}}$. Asymptotically with the code length, the consistent Maximum Likelihood Estimator (MLE) parameterized on $c_{\text{max}}$ as defined in (18), yields an estimation $\hat{\theta}_{c_{\text{max}}} \approx \theta_{c_{\text{max}}} \in \mathcal{E}(\Theta_c)$ with increasing accuracy. This is not an estimation of the true collusion attack because $c \neq c_{\text{max}}$ a priori. Therefore, we prefer to refer to $\hat{\theta}_{c_{\text{max}}}$ as a collusion inference, and the scoring uses this inference as follows:

$$s_j = \sum_{i=1}^m \log \frac{P(y(i)|x_j(i), \hat{\theta}_{c_{\text{max}}})}{P(y(i)|\hat{\theta}_{c_{\text{max}}})}.$$  

Suppose that the MLE tuned on $c_{\text{max}}$ provides a perfect inference $\hat{\theta}_{c_{\text{max}}} = \theta_{c_{\text{max}}}$, we then succeed to restrict the compound channel to the discrete set $\mathcal{E}_{c_{\text{max}}} = \mathcal{E}(\Theta) \cap \mathcal{E}(\Theta_{c_{\text{max}}})$ which we define as the restriction of $\mathcal{E}(\Theta)$ to collisions of size $\tilde{c} \leq c_{\text{max}}$. Appendix A shows that $\mathcal{E}_{c_{\text{max}}} = \mathcal{E}(\Theta)$ is one-sided w.r.t. $f = f_T$, and its worst attack is indeed $\theta_{c_{\text{max}}}$. Lemma 5 of [5] justifies the use of the MAP decoder (4) tuned on $\theta_{c_{\text{max}}}$. Its application leads to a more efficient decoder since $R_S(f_T, \theta_{c_{\text{max}}}) > R_S(f_T, \hat{\theta}_{c_{\text{max}}}, f_T)$. This decoder pertains to the approach based on score adaptation, with the noticeable advantages: it is better theoretically grounded and it is far less complex than the iterative E.-M. decoder of [10].

Figure 2 illustrates the Detection Error Trade-off (DET) per user for the single decoders discussed so far with $m = 512$ and $c = 5$ colluders performing worst-case (i.e. minimizing $R_S(f_T, \theta)$ over $\Theta_S$) and majority attack ($\theta_{5, \text{maj}} = \ldots$).
transmitted, whereas in fingerprinting, there is a unique codeword for a given couple \((p, y)\) and a known \(n\). Once these are fixed, the score \(s_j = s(x_j, y, p)\) is a deterministic function from \(\{0, 1\}^m\) to \(\mathbb{R}\). Since the codewords of the innocent users are i.i.d. and \(c \ll n\), we have:

\[
P_{fp} = 1 - (1 - \mathbb{P}(s(x_{inn}, y, p) > \tau))^n - c
\]

The number of possible codewords can be evaluated as the number of typical sequences, i.e., in the order of \(2^{n\mathbb{E}_{p,y}[\log_2 p]}\), with \(h_0(p)\) the entropy in bits of a Bernoulli random variable \(B(p)\), \(\mathbb{E}_{p,y}[\log_2 p] \approx 0.557\) bits, which leads to a far bigger number of typical sequences than \(n\) (say \(m \geq 300\) and \(n \leq 10^8\) in practice). This shows that plenty of codewords have not been created when a pirate copy is found. Therefore, we consider them as occurrences of \(x_{inn}\) since we are sure that they have not participated in the forgery of \(y\). The idea is then to estimate \(\tau\) s.t. \(\mathbb{P}(s(x_{inn}, y, p) > \tau) = n^{-1}P_{fp}\) with, for instance, a Monte Carlo simulation with newly created codewords.

The difficulty lies in the order of magnitude. Some typical requirements are \(n \approx 10^6\) and \(P_{fp} = 10^{-4}\), hence an estimation of \(\tau\) corresponding to a probability as small as \(\pi = 10^{-10}\). This is not tractable with a basic Monte Carlo on a regular computer because it requires \(O(\pi^{-1})\) runs. However, the new estimator based on rare event analysis proposed in [25] performs remarkably fast within this range of magnitude. It produces \(\hat{\tau}\) and a \(C\%-\) confidence interval\(^2\) \([\hat{\tau} - \tau^\pm, \hat{\tau} + \tau^\pm]\) with only \(O(\log(\pi^{-1}))\) runs. In our decoder, we compare the scores to \(\tau^+\) (i.e., a pessimistic estimate of \(\tau\)) to ensure a total false positive probability lower than \(P_{fp}\). Last but not least, this approach works for any deterministic scoring function \(s(\cdot,\cdot)\), and is also applied to joint decoding in Sect. IV-C.

IV. ITERATIVE, JOINT DECODING ALGORITHM

This section extends the single decoder based on the collusion inference \(\theta_{c_{max}}\) towards joint decoding, according to the guidelines of Sec. II-E. Preliminary results about these key ideas were first presented in [26] and [27]. The description below makes references to the lines of the pseudo code of Algorithm 1.

A. Architecture

The first principle is to iterate the score computation and include users accused in previous iterations as side-information to build a more discriminatory test. Let \(U_{SI} = \emptyset\) denote the initially empty set of accused users (line 1). In each iteration we aim at identifying a (possibly empty) set of users \(A\) (lines 8 & 20) and then update \(U_{SI}\) with \(A\) (line 24).

\(^2\)In practice, we set \(C = 95\), i.e. we are 95%-sure that the true \(\tau\) lies in this interval.
Second, we additionally compute scores for subsets of \( t \) users of \( \mathcal{U} \setminus \mathcal{U}_S \), \( t \leq c_{\text{max}} \) (line 13). There are \( \binom{t}{t} \cdot t_{\text{max}} \) such subsets. As \( n \) is large, enumerating and computing a score for each subset is intractable even for small \( t \). The idea here is to find a restricted set \( \mathcal{U}^{(t)} \subseteq \mathcal{U} \setminus \mathcal{U}_S \) of \( n^{(t)} = |\mathcal{U}^{(t)}| \) users (line 11) that are the most likely to be guilty and to keep \( p^{(t)}(i) = \frac{1}{n^{(t)}} \) approximately constant from one iteration to another and within our computation resources. We gradually reduce \( n^{(t)} \) by pruning out users who are unlikely to have taken part in the collusion when going from single (\( t = 1 \)) decoding, to pair (\( t = 2 \)) decoding, etc. If \( n^{(t)} = O(n^{(t+1)}) \), then score computation of \( t \)-subsets over the restricted user set is within \( O(n) \) just like for the single decoder. By abuse of notation, \( \binom{\mathcal{U}^{(t)}}{t} \) denotes the set of all \( t \)-subsets of \( \mathcal{U}^{(t)} \) (line 13).

Initially, the single decoder computes the score of all users (line 6) and accuses those whose score is above the estimated threshold \( \tau^+ \) (line 8). If this happens, these users are included in \( \mathcal{U}_S \) (line 24), and the single decoder restarts by better estimating the collusion inference and computing more discriminative scores thanks to the side-information. If nobody is accused, we rank the users according to their ‘single’ score, i.e. the top-ranked user is most likely to be a colluder, and only the \( n^{(2)} \) first users are included in \( \mathcal{U}^{(2)} \). This list of suspects is passed to the joint decoder for \( t = 2 \), i.e. a pair decoder.

The joint decoder produces a new list of scores computed from subsets of \( t \) users (line 13), which – according to theoretical results [1], [3] – are more discriminative as \( t \) increases. Yet, the accusation and the pruning operations at this stage are more involved than with the single decoder. Denote \( \mathcal{T}^0 \subseteq \mathcal{U}^{(t)} \) the \( t \)-subset of users with the highest score (line 15). If this score is above the threshold, the joint decoder tries to accuse the most likely colluder within \( \mathcal{T}^0 \) (lines 17–21 – see Sect. IV-C). Therefore at most one user is accused at this step (contrary to the single decoder, which may accuse more than one user). If this happens, \( \mathcal{U}_S \) is updated (line 24), and the single decoder restarts taking into account this new side-information. If no accusation can be made by the joint \( t \)-subset decoder, we generate a new and shorter list of suspects \( \mathcal{U}^{(t+1)} \) based on the ranking of joint scores (line 11) that is fed to the subsequent \( (t+1) \) joint decoding stage (see Sect. IV-C).

In the detect-one scenario, the algorithm stops after the first accusation. We restrict the subset size to \( t \leq t_{\text{max}} \), with \( t_{\text{max}} = 5 \). This is not a severe limitation as for moderately large \( c \), the decoding performance advantage of the joint decoder quickly vanishes [1]. In the detect-many scenario, iteration stops when \( |\mathcal{U}_S| \geq c_{\text{max}} \) or \( t \) reaches \( \min(t_{\text{max}}, c_{\text{max}}-|\mathcal{U}_S|) \) and no further accusation can be made. The set \( \mathcal{U}_S \) then contains the user indices to be accused. Alg. 1 gives the architecture of the accusation process for the catch-many scenario.

The next sections describe the score computation, the pruning and the accusation, and the inference of the collusion process in more detail.

### B. Score computation

This section extends the scoring (10) of the single decoder to joint decoding. Denote by \( \Xi_\mathcal{E} \) the set of codewords of the users of set \( \mathcal{E} \). For a \( t \)-subset \( \mathcal{T} \), the accusation is formulated as a hypothesis test based on the observations \( (\Xi_\mathcal{T}, y, \mathbf{p}) \) and on the side-information \( \mathcal{U}_S \) to decide between \( H_0 \) (all \( j \in \mathcal{T} \) are innocent) and \( H_1 \) (all \( j \in \mathcal{T} \) are guilty). The joint score of subset \( \mathcal{T} \), \( s_\mathcal{T} = s(\Xi_\mathcal{T}, y, \mathbf{p}, \Xi_\mathcal{U}_S) \), is just the LLR tuned on the inference \( \theta_{\text{max}} \) of the collusion process. This description encompasses single scores (10) for \( t = 1 \) (lines 6 & 20) and joint scores for \( t > 1 \) (line 13). In the latter case, the alternative hypothesis \( H_1 \) should indeed be: there is at least one \( j \in \mathcal{T} \) who is guilty. But this composite hypothesis test has a complexity in \( O(2^t) \) per \( t \)-subset. Our approach is suboptimal for \( t > 1 \) but less complex.

The sequences \( \mathbf{p}, y \) and the codewords of the codebook \( \Xi_\mathcal{E} \) are composed of independent random variables thanks to the code construction and the memoryless nature of the collusion. Moreover, the collusion only depends on the number of ‘1’ symbols present in the codewords of a subset. Denote by \( \varphi \) and \( \delta \) the accumulated codewords corresponding to \( \mathcal{T} \) and \( \mathcal{U}_S \): \( \varphi = \sum_{j \in \mathcal{T}} x_j \) and \( \delta = \sum_{j \in \mathcal{U}_S} x_j \). We have \( \forall i \in [m], 0 \leq \varphi(i) \leq t \) and \( 0 \leq \delta(i) \leq n_{\text{si}} \), where \( n_{\text{si}} = |\mathcal{U}_S| \). Thanks to the linearity structure of the decoder, the score for a subset \( \mathcal{T} \) of \( t \) users is simply:

\[
s_\mathcal{T} = \sum_{i=1}^m W(\varphi(i), i),
\]

where \( W(j,i) \) is the entry in row \( j \) and column \( i \) of a \((t+1) \times m\) weight matrix \( W \) which is pre-computed (procedure

---

### Algorithm 1 Iterative Joint Tardos Decoder

Require: \( y, \Xi, \mathbf{p}, c_{\text{max}}, t_{\text{max}} \leq c_{\text{max}}, n^{(t)}, P_p \\
1: \mathcal{U} = \{j | 1 \leq j \leq n\}, \mathcal{U}_S = \emptyset \\
2: \text{repeat} \\
3: \quad t = 1 \\
4: \quad \theta_{\text{max}} = \text{infer}(\mathbf{y}, \mathbf{p}, \mathcal{U}_S, c_{\text{max}}) \\
5: \quad W = \text{weights}(\mathbf{y}, \mathbf{p}, \theta_{\text{max}}) \\
6: \quad s = \text{scores}(\mathcal{U} \setminus \mathcal{U}_S, \Xi, W) \\
7: \quad \tau^+ = \text{threshold}(\mathbf{p}, W, P_p, n, t) \\
8: \quad \mathcal{A} = \{j \in \mathcal{U} \setminus \mathcal{U}_S | s_j > \tau^+ \} \\
9: \quad \text{while } \mathcal{A} = \emptyset \text{ and } t < \min(t_{\text{max}}, c_{\text{max}} - |\mathcal{U}_S|) \text{ do} \\
10: \quad \quad t = t + 1 \\
11: \quad \quad \mathcal{U}^{(t)} = \text{top}(s, \mathcal{U} \setminus \mathcal{U}_S, n^{(t)}) \\
12: \quad \quad W = \text{weights}(\mathbf{y}, \mathbf{p}, \theta_{\text{max}}) \\
13: \quad \quad s = \text{scores}(\mathcal{U}^{(t)} \setminus \Xi, W) \\
14: \quad \quad \tau^+ = \text{threshold}(\mathbf{p}, W, P_p, n, t) \\
15: \quad \quad \Sigma^0 = \text{arg max}_{\mathcal{T} \subseteq \mathcal{U}^{(t)} \setminus \Xi} |\mathcal{T}| \\
16: \quad \quad \text{if } s_{\mathcal{T}} > \tau^+ \text{ then} \\
17: \quad \quad \quad \text{for all } j \in \mathcal{T} \text{ and } \mathcal{A} = \emptyset \text{ do} \\
18: \quad \quad \quad \quad W = \text{weights}(\mathbf{y}, \mathbf{p}, \theta_{\text{max}}) \cup \{\mathcal{T} \setminus \{j\}\} \\
19: \quad \quad \quad \quad \tau^+ = \text{threshold}(\mathbf{p}, W, P_p, n, t) \\
20: \quad \quad \quad \quad \text{if score}(\{j\}, \Xi, W) > \tau^+ \text{ then } \mathcal{A} = \{j\} \\
21: \quad \quad \text{end for} \\
22: \quad \quad \text{end if} \\
23: \quad \quad \text{end while} \\
24: \quad \mathcal{U}_S = \mathcal{U}_S \cup \mathcal{A} \\
25: \quad \text{until } \mathcal{A} = \emptyset \text{ or } |\mathcal{U}_S| \geq c_{\text{max}} \\
26: \quad \text{return } \mathcal{U}_S \]
weights() in Alg. 1) from \((y, p)\) taking into account the side 
information \(L_{SI}\) so that \(\forall (\varphi, i) \in \{0, \ldots, t\} \times \{1, \ldots, m\}: \)
\[
W(\varphi, i) = \log \frac{P(y(i)|(\varphi, t), (\delta(i), n_{SI}), p(i), \hat{\theta}_{c_{\max}})}{P(y(i)|(\delta(i), n_{SI}), p(i), \hat{\theta}_{c_{\max}})}.
\]
(15) 
For indices s.t. \(y(i) = 1\), both the numerator and the denominator 
share a generic formula, \(P(\varphi(i)+\delta(i), t+n_{SI}, p(i), \hat{\theta}_{c_{\max}})\) 
and \(P(\delta(i), n_{SI}, p(i), \hat{\theta}_{c_{\max}})\) respectively, with 
\[
P(u, v, p, \hat{\theta}_{c_{\max}}) = \sum_{k=u}^{c_{\max} - v - u + 1} \left( \hat{\theta}_{c_{\max}}(k) \right) \left( c_{\max} - v \right) \left( k - u \right) \left( 1 - p \right)^{k - u} \left( 1 - \frac{\delta}{c_{\max} - v - k + u} \right).
\]
(16) 
In words, this expression gives the probability that \(y = 1\) 
knowing that the symbol ‘1’ has been distributed to users with 
probability \(p\), the collusion model \(\hat{\theta}_{c_{\max}}\), and the identity 
of \(v\) colluders who have \(u\) symbols ‘1’ and \(v - u\) symbols ‘0’. 
For indices s.t. \(y(i) = 0\) in (15), the numerator and the 
denominator need to be ‘mirrored’: \((P^\dagger \to 1 - P)\). 
At iterations based on the single decoder (lines 6 & 20), \(t = 1\) 
and \(\varphi = x_j\) for user \(j\). If nobody has been deemed guilty 
so far, then \(\delta(i) = n_{SI} = 0, \forall i \in [m]\). This score is defined if 
\(t + n_{SI} \leq c_{\max}\). Therefore, for a given size of side-information, 
we cannot conceive a score for subsets of size bigger than 
\(c_{\max} - n_{SI}\). This implies that in the detect-many scenario, 
the maximal number of iterations depends on how fast \(L_{SI}\) grows. 
The procedure scores() in Alg. 1 outputs a list of scores 
given a set of subsets of \(t\) users (which is the set of users if 
\(t = 1\) for single decoding), the code matrix and the weight 
matrix \(W\). We assume there is a deterministic way to 
choose all the \(t\)-subsets; in practice, this is done by the revolving 
door procedure [27]. 

C. Ranking users and accusation 
In order to build the set \(I^{(t)}\), we need to rank the users 
based on the previous scores. We record the highest score for 
each user: 
\[
s_j = \max_{T_j \in \mathcal{T}} s_{T_j}.
\]
(17) 
This step is not needed if the scores come from a single 
decoder. The procedure top() (line 11) ranks users 
according to their highest subset score and prunes the suspect list to the 
first \(n^{(t)}\) users. 
Suppose \(T_{mn}\) is a \(t\)-subset composed of innocent users. 
Using rare event analysis, \(\tau^+\) is a pessimistic estimation of \(\tau\) 
(see Sect. III-C) s.t. \(\mathbb{P}(s(X_{mn}, y, p, \Xi_{L_{SI}}(T\setminus\{j\})) > \tau^+) \leq \binom{n}{t}^{-1} P_{fp}\). 
This is the procedure threshold(p, W, P_{fp}, n, t) applied 
with \(t = 1\) (lines 7 & 19) or \(t \geq 1\) (line 14). Let \(\mathcal{T}\) denote 
the \(t\)-subset with the highest score (line 15). If \(s_{\mathcal{T}} > \tau^+\), 
then \(\mathcal{T}\) contains at least one colluder with a high probability. 
This holds for any scoring function, and especially with the 
one explained in Sect. IV-B even if it is not optimal. 
We accuse at most one user in \(\mathcal{T}\). We compute the 
following single score \(s(s_j, y, p, \Xi_{L_{SI}}(T\setminus\{j\}))\), and we 
accuse user \(j\) if it is bigger than \(\tau^+\), with \(\tau^+\) s.t. 
\[
\mathbb{P}(s(X_{mn}, y, p, \Xi_{L_{SI}}(T\setminus\{j\})) > \tau^+) \leq n^{-1} P_{fp}\ (\text{line 19}).
\]

D. Inference of the collusion process 
The MLE infers about the collusion process (line 4): 
\[
\hat{\theta}_{c_{\max}} = \arg \max_{\theta \in \Theta_{c_{\max}}} \mathbb{P}(y|p, L_{SI}, \theta).
\]
(18) 
Whenever a user is deemed guilty, the user is added to side-
information and we re-run the parameter estimation to refine 
the collusion inference. Our soft decision decoding method 
resorts to the noise-aware collusion models (2) and (3) and sets 
\[
\hat{\theta}_{c_{\max}} = \arg \max_{\theta \in \Theta_{c_{\max}}} \mathbb{P}(y'|p, L_{SI}, \theta).
\]
(19) 
This is illustrated in the right part of Fig. 3. Notice that models 
I and II share the same number of parameters, therefore, 
there is no risk of over-fitting. Replacing (18) by (19) in the 
collusion inference step is the only change to the decoding 
algorithm we make to handle collusion attacks over real 

V. EXPERIMENTAL RESULTS 
We have implemented the Tardos decoders in C++. Single 
and joint score computation is implemented efficiently using 
pre-computed lookup tables, cf. (14) and (15), and aggregation 
techniques described in [26]. For a code length of \(m = 1024\) 
more than \(10^6\) single and about \(10^7\) joint scores, respectively, 
can be computed per second on a single core of a regular 
Intel Core2 2.6 GHz CPU. To control the runtime, the joint 
decoders are confined to \(s\)-subset decoding (\(t_{max} = 5\)) and 
around \(p^{(t)} \approx 4.5 \cdot 10^6\) computed subsets per joint decoding 
stage. An iterative decoding experiment can be executed on a 
PC within a couple of minutes, given enough memory, see 
[27] for details. To experimentally verify the false-positive 
rate controlled by rare-event analysis, up to \(3 \cdot 10^4\) tests per 
parameter setting have been performed on a cluster of PCs. 
First, we compare the performance of the proposed decoders 
under marking assumption. Finally, we lift this unrealistic 
restriction and turn to a more practical assessment using soft-
decision decoding. 

Unless explicitly noted, the terms single and joint 
decoder refer to the decoders conditioned on the inference of 

1Source code is available at http://www.irisa.fr/_texmex/people/furon/sec. html.
the collusion process $\hat{\theta}_{\text{max}}$, cf. (10) and (14). Further, we consider the MAP decoders assuming knowledge of $\theta_e$ and the compound channel decoder, cf. (9), tuned on the worst-case attack $\theta^*_k, \forall k \in [2, \ldots, c_{\text{max}}]$. As a baseline for a performance comparison, we always include symmetric Tardos score computation [9] with a threshold controlled by rare-event analysis (see Sec. III-C).

A. Decoding performance under marking assumption

1) Detect-one scenario: Here the aim is to catch at most one colluder – this is the tracing scenario most commonly considered in the literature. We compare our single and joint decoder performance against the results provided by Nuida et al. [13] (which are the best as far as we know) and, as a second reference, the symmetric Tardos decoder.

The experimental setup considers $n = 10^6$ users and $c \in \{2, 3, 4, 6, 8\}$ colluders performing worst-case attack [22] against a single decoder. In Fig. 4, we plot the empirical probability of error $P_e = P_{\text{fp}} + P_{\text{fn}}$ obtained by running $10^3$ experiments for each setting versus the code length $m$. The false-positive error is controlled by thresholding based on rare-event simulation, $P_{\text{fp}} = 10^{-3}$, which is confirmed experimentally. Evidently, for a given probability of error, the joint decoder succeeds in reducing the required code length over the single decoder, especially for larger collusions.

Table I compares the code length to obtain an error rate of $P_e = 10^{-3}$ for our proposed Tardos decoders and the symmetric Tardos decoder with the results reported by Nuida et al. [13, Table 4] under marking assumption. Except for $c = 2$, the proposed decoders can substantially reduce the required code length and the joint decoder improves the results of the single decoder. Note that Nuida’s results give analytic code length assuming a particular number of colluders for constructing the code but the collusion attack is arbitrary (i.e. not necessary fair) whereas our results are experimental estimates based on worst-case attack against a single decoder and without knowing $c$ at the code construction. Results with $c$ known are provided in [27] and show a slightly better performance: the required code length of the joint decoder is then slightly shorter than Nuida’s code in case $c = 2$.

2) Detect-many scenario: We now consider the more realistic case where the code length $m$ is fixed and the false-negative error rate is only a minor concern$^4$ while the false-positive probability is critical to avoid an accusation of an innocent. The aim is to identify as many colluders as possible.

Figures 5(a)–5(d) show the average number of identified colluders by different decoding approaches. The experimental setup considers $n = 10^6$ users, code length $m = 2048$, and several collusion attacks (worst-case, i.e. minimizing the achievable rate of a single or joint decoder, interleaving and majority which is a rather mild attack) carried out by two to eight colluders. The global probability of a false positive error is fixed to $P_{\text{fp}} = 10^{-3}$.

As expected, the MAP single decoder knowing $\theta_e$ provides the best decoding performance amongst the single decoders, yet is unobtainable in practice. The symmetric Tardos decoder performs poorly but evenly against all attacks; the single decoder based on the compound channel (9) improves the results only slightly.

The joint decoders consistently achieve to identify most colluders – with a dramatic margin in case the traitors choose the worst-case attack against a single decoder. This attack bothers the very first step of our decoder, but as soon as some side information is available or a joint decoder is used, this is no longer the worst case attack. Finding the worst case attack against our iterative decoder is indeed difficult. A good guess is the interleaving attack which is asymptotically the worst case against the joint decoder [2]. The experiments show that it reduces the performance of the joint decoders substantially for large $c$.

The decoder based on the inference $\hat{\theta}_{\text{max}}$ and the true MAP are different when $c$ is lower than $c_{\text{max}}$. However, this is not a great concern in practice for a fixed $m$: for small $c$, the code is long enough to face the collusion even if the score is less discriminative than the ideal MAP; for big $c$ the score of our decoder gets closer to the ideal MAP.

B. Decoding performance of the soft decoder

We assess the performance of the soft decision decoders proposed in Sec. II-C in two tracing scenarios: (i) Kuribayashi considers in [14] $n = 10^4$ users and code length $m = 10^4$, (ii) a large-scale setup with 33,554,432 users and $m = 7,440$ where Joutras and Moulin [16] provide results for their high-rate random-like fingerprinting code under averaging and interleaving attack.

In Fig. 6, we compare the average number of identified colluders for the single and joint decoder using different estimates

---

4A tracing schemes rightly accusing a colluder half of the time might be enough to dissuade dishonest users.
of the collusion process. A simple approach, termed hard decision decoding in the sequel, first thresholds \( y'(i) \) (to quantize \( y'(i) \) into 0 if \( y'(i) < 0 \) and 1 otherwise), and then employs the collusion process inference \( \hat{\theta}_{\text{max}} \) of (18) on the hard outputs. The term soft relates to the noise-aware decoders relying on \( \hat{\theta}^{(J)}_{\text{max}} \) or \( \hat{\theta}^{(F)}_{\text{max}} \) chosen adaptively based on the likelihood of the two models (see (19)). All plots also show the results for the (hard-thresholding) symmetric Tardos decoder. The false-positive rate is set to \( 10^{-4} \). Extensive experiments (\( 3 \times 10^4 \) test runs) have been carried out to validate the accusation thresholds proposed in [14] with a rare-event simulation, we are able to fix the false-alarm rate in case of the symmetric Tardos code. Furthermore, the decoding results given in [14] for the discrete variant of the fingerprinting code (i.e., Nuida’s construction) could be significantly improved by rare-event simulation based thresholding. Contrary to the claim of [14], soft decision decoding always provides a performance benefit over the hard decoders.

In Fig. 7 we illustrate the decoding performance when dealing with a large user base. We consider averaging and interleaving attacks by \( c = 2, \ldots, 12 \) and \( c = 2, \ldots, 8 \) colluders (\( c_{\text{max}} = 12 \) and \( c_{\text{max}} = 8 \), respectively) followed by AWGN with variance \( \sigma_n^2 = 1 \). The global false-positive rate is set to \( 10^{-3} \). The benefit of the soft decoding approach is clearly evident. Joint decoding provides only a very limited increase in the number of identified colluders. For comparison, Jourdas & Moulin indicate an error rate of \( P_e = 0.0074 \) for

![Fig. 5. Decoder comparison in the detect-many tracing scenario: \( n = 10^6 \), \( n = 2048 \), \( P_{\text{fp}} = 10^{-3} \), \( c_{\text{max}} = 8 \). (Best viewed in color.)](image)

![Fig. 6. Kuribayashi setup: \( n = 10^4 \), \( n = 10^4 \), \( P_{\text{fp}} = 10^{-4} \), \( c = 10 \), \( c_{\text{max}} = 20 \); worst-case, interleaving, majority and averaging attack followed by AWGN (\(-4, \ldots, 10 \) dB SNR).](image)
Table 1 shows the average number of identified colluders for different settings:

<table>
<thead>
<tr>
<th>Number of Colluders (c)</th>
<th>Single (Hard)</th>
<th>Joint (Hard)</th>
<th>Single (Soft)</th>
<th>Joint (Soft)</th>
<th>Symm. Tardos</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>2</td>
<td>7</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
</tbody>
</table>

In the first setting for a detect-one scenario [16], the false-positive rate is given for several collusion channels. The decoder achieves a competitive result: with the given code length, all colluders in the second setting for a detect-one scenario [16].

In [28], $P_{bp} = 0.0016$ and $P_{bn} = 0.044$ are given for the first experiment (Fig. 7(a)) by introducing a threshold to control the false-positive rate. Our soft joint decoder achieves a $P_{bn} = 0.046$ for $P_{bp} = 10^{-3}$ (for $c = 10$ colluders), catching 2.6 traitors on average.

In the second experiment (see Fig. 7(b)), our joint decoder compares more favorably: with the given code length, all $c = 5$ colluders can be identified and for a collusion size $c = 8$, 4.5 traitors are accused without observing any decoding failure in $3 \cdot 10^3$ tests.

C. Runtime Analysis

Single decoding can be efficiently implemented to compute more than one million scores for a code of length $m = 1024$ per second. Its complexity is in $O(m \cdot n)$. Selecting the $n^{(1)}$ most likely guilty users can be efficiently done with the maxheap algorithm. Yet, it consumes a substantial part of the runtime for small $m$. The runtime contribution of score computation for the joint decoding stages is in $O(m \cdot p^{(1)})$ and clearly depends on the size of the pruned list of suspects. However, the computation of the score per subset is independent of the subset size $t$ thanks to the revolving door enumeration method of the subsets $\theta$. Restricting $p^{(1)}$ and $t_{\max}$ keeps the joint decoding approach computationally tractable. Better decoding performance can be obtained using higher values at the cost of a substantial increase in runtime. Experiments have shown that even the moderate settings ($p^{(1)} \approx 4.5 \cdot 10^6$ and $t_{\max} = 5$) achieve a considerable gain of the joint over the single decoder for several collusion channels.

Thresholding accounts for more than half of the runtime in the experimental setups investigated in this work. However, this is not a serious issue for applications with a large user base or when $p^{(1)}$ becomes large. Thresholding depends on the subset size $t$ because a large number of random codeword combinations must be generated and because we seek lower probability level in $O(P_{bp}/n^t)$. Therefore, the complexity is in $O(m \cdot t^2 \cdot \log n)$ according to [25]. There are no more than $c_{\max}$ such iteration with $t \leq c_{\max}$, so that the complexity of the thresholding is in $O(m \cdot \log n)$ and the global complexity of our decoder stays in $O(m \cdot n)$.

More details about the runtime of our implementation are given in [27]. Note that results have been obtained with a single CPU core although a parallel implementation can be easily achieved.

VI. Conclusion

Decoding fingerprinting codes in practice means to trace guilty persons over a large set of users while having no information about the size nor the strategy of the collusion. This must be done reliably by guaranteeing a controlled probability of false alarm.

Our decoder implements provably good concepts of information theory (joint decoding, side information, linear decoder for compound channels) and statistics (estimation of extreme quantile of a rare event). Its extension to soft output decoding is straightforward as it does not change the architecture. Very competitive results have been obtained experimentally.

Since the proposed iterative method is neither just a single decoder nor completely a joint decoder (it only considers subsets over a short list of suspects), it is rather difficult to find the best distribution for code construction and its worst case attack. Experiments show that the interleaving attack is indeed more dangerous than the worst-case attack against a single decoder.

ACKNOWLEDGMENT

This research was partially supported by the French ANR project Medievals. The authors are grateful for the efforts of the anonymous reviewers who contributed by their valuable comments to the quality of this article.

APPENDIX

We prove that $E_{\mathcal{C}_{\text{hyp}}} (\theta_k) \triangleq \{\tilde{\theta}_k | k \leq c_{\max}, \mathbb{P}(y|p, \tilde{\theta}_k) = \mathbb{P}(y|p, \theta_\ast), \forall (y, p) \in [0, 1] \times [0, 1]\}$ is one sided w.r.t. some $p \in [0, 1]$ and thus w.r.t. the expectation over $f$ if $f(p) > 0$ for at least one of these values of $p$. The collusion channels of this set share the property that $\mathbb{P}(Y = 1|p, \tilde{\theta}_k) = q(p) \geq 0, \forall p \in [0, 1]$. From [22, Eq. (20)]:

$$
\mathbb{P}(Y = 1|X = 1, p, \tilde{\theta}_k) = q(p) + k^{-1}(1 - p)q'/(p)
$$

$$
\mathbb{P}(Y = 1|X = 0, p, \tilde{\theta}_k) = q(p) - k^{-1}pq'(p)
$$

Taking $(\tilde{\theta}_k, \tilde{\theta}_b) \in \mathcal{C}_{\text{hyp}} (\theta_\ast)^2$ s.t. $k_A < k_B$. We first show that $R(f, \theta_\ast) \geq R(f, \tilde{\theta}_k)$ so that $\theta_\ast$ is a minimizer of $R(f, \theta)$ over $\mathcal{C}_{\text{hyp}} (\theta_\ast)$. Denote by $(\mu_1, \mu_2)$ the following conditional probability distributions:

$$
\mu_1(y|X, p) = \mathbb{P}(Y = y|p) = q(p)^y(1 - q(p))^{1 - y}
$$

$$
\mu_2(y|X, p) = \mathbb{P}(Y = y|X, p, \tilde{\theta}_k).
$$

Then, $\mathbb{P}(Y|X, p, \tilde{\theta}_k) = (1 - \lambda)\mu_1(Y, X|p) + \lambda\mu_2(Y, X|p)$, $\forall p \in [0, 1]$, with $\lambda = k_A/k_B < 1$. The mutual information

---

**Fig. 7.** Jourdas & Moulin setup: $n = 33554432$, $m = 7440$, $P_{bp} = 10^{-3}$, averaging and interleaving attack followed by AWGN (0 dB SNR).
is a convex function of $\mathbb{P}(Y|X,p)$ for fixed $\mathbb{P}(X|p)$ so that, once integrated over $f(p)$, we have

$$R(f,\theta_A) \leq (1 - \lambda) \cdot 0 + \lambda \cdot R(f,\theta_{c_A}) \leq R(f,\theta_{c_A}). \tag{24}$$

The second inequality turns to be an equality if only if $R(f,\theta_{c_A}) = 0$. It means that $k_A$ colluders succeed to nullify the mutual information between $X$ and $Y$ for any $p$ s.t. $f(p) > 0$. Then, $\theta_{c_A}$ is not a unique minimizer. This can happen if $k_A$ is big enough, but it is impossible for distributions s.t. $f(p) > 0$ for some $p < 1/k_A$, see [22, Sec. 4]. This especially holds for $f_T$.

We now prove that (8) holds $\forall \theta \in \mathcal{E}_{\max}(\theta_c)$. This is equivalent to

$$R(f,\theta_T) - D(\mathbb{P}(Y|X,\theta_T)||\mathbb{P}(Y|X,\theta_{c_A})) - R(f,\theta_{c_A}) \geq 0, \tag{25}$$

where the LHS is of the form $\mathbb{E}_{p \sim f}[g(P)]$ with $g(0) = g(1) = 0$. After developing the expressions for $0 < p < 1$, we find that:

$$g(p) = \left( k^{-1} - c_{\max}^{-1} \right) p(1-p) \cdot \left( q'(p) \log \left( 1 + \frac{1 - p}{c_{\max}} \frac{q'(p)}{q(p)} \right) + q'(p) \log \left( 1 - \frac{1 - p}{c_{\max}} \frac{1 - q'(p)}{1 - q(p)} \right) - q'(p) \log \left( 1 - \frac{1 - p}{c_{\max}} \frac{q'(p)}{q(p)} \right) \right). \tag{26}$$

The four terms inside parenthesis are not negative because $x \log(1 + \gamma x) \geq 0$ for $\gamma > 0$ and $x > -\gamma^{-1}$. Since $k < c_{\max}$, we obtain $g(p) \geq 0$ and (8) after expectation over $f$.

REFERENCES


Peter Meerwald received the Master degree in Computer Sciences from Bowling Green State University (Ohio, USA) in 1999 and obtained a Master and PhD degree from the University of Salzburg (Austria) in 2001 and 2010.

From 2001 to 2007, he was a Research Engineer with Sony DADC (Salzburg, Austria) working on software security and copy protection. In 2010/2011, he pursued postdoctoral research on multimedia fingerprinting at Inria Research Center (Rennes, France). At present, he is a Research Engineer with bt electronic (Salzburg, Austria) developing audio processing and communication systems on embedded Linux platforms.

Teddy Furon received the M.S. degree in digital communications and the Ph.D. degree in signal and image processing from the Ecole Nationale Supérieure des Télécommunications de Paris, Paris, France, in 1998 and 2002, respectively.

From 1998 to 2001, he was a Research Engineer with the Security Lab of THOMSON, Rennes, France, working on digital watermarking in the framework of copy protection. He continued working on digital watermarking as a Postdoctoral Researcher at the TELE Lab of the Université Catholique de Louvain, Louvain, Belgium. He also worked in the Security Lab of Technicolor. He is at present a researcher working within the TEXMEX Team-Project in the Inria Research Center, Rennes, France.