Efficient Watermark Detection and Collusion Security

Francis Zane

Department of Fundamental Mathematics
Bell Laboratories
700 Mountain Avenue
Murray Hill, NJ 07974
E-mail: francis@research.bell-labs.com
URL: http://cm.bell-labs.com/cm/ms/who/francis

Abstract. Watermarking techniques allow the tracing of pirated copies of data by modifying each copy as it is distributed, embedding hidden information into the data which identifies the owner of that copy. The owner of the original data can then identify the source of a pirated copy by reading out the hidden information present in that copy. Naturally, one would like these schemes to be as efficient as possible. Previous analyses measured efficiency in terms of the amount of data needed to allow many different copies to be distributed; in order to hide enough data to distinguish many users, the total original data must be sufficiently large. Here, we consider a different notion of efficiency: What resources does the watermark detector need in order to perform this tracing?
We address this question in two ways. First, we present a modified version of the CKLS media watermarking algorithm which improves the detector running time from linear to polylogarithmic in the number of users while still maintaining collusion-security. Second, we show that any public, invertible watermarking scheme secure against c colluding adversaries must have at least $\Omega(c)$ bits of secret information.

1 Introduction

The problem of preventing piracy, particularly of multimedia data, is gaining new importance as improvements in compression and bandwidth allow the efficient redistribution of copied data on a wide scale. One technique used to discourage copying is *watermarking* (also known as *fingerprinting*): distributing a distinct, modified copy of the data to each user to allow tracing of pirated copies to their original owners. In developing watermarking schemes, the goal is to allow many copies of the data to be distributed while maintaining this traceability property. Naturally, the pirates will behave in an adversarial fashion, using whatever information is at their disposal in order to evade detection. In addition to attacks which operate on a single copy, the pirates may collude, combining information from many copies. Our focus in this paper is on improving the efficiency of

schemes which allow the tracing of copies even in the presence of colluding adversaries, and in particular on improving the running time of watermark decoding algorithms.

At an abstract level, most media watermarking schemes follow the same pattern: For each user, a individual noise pattern (called a watermark) is generated and inserted into the document given to him. Each suspect document is compared to the watermarks distributed; if a user's watermark matches the one in the document, that user is incriminated. Generating these watermarks in a manner which does not interfere with the original data yet is hard to remove is the core of any watermarking scheme.

A natural strategy is to fix some distribution and generate these noise patterns randomly and independently for each user. This strategy leads to schemes which are amenable to analysis. At an intuitive level, implicating an innocent user requires guessing (at least approximately) the watermark corresponding to that user, which is very unlikely. At the same time, by analyzing the chosen distribution, one can quantify how much information about the original data the adversary obtains by looking at a set of watermarked copies. This intuition is the core of the analysis presented in [6] of the CKLS watermarking scheme [3] as well as the random coding or random hash function argument used in [2] and [1]. All these schemes make use of random codes to encode the identities of many users into relatively short documents in a collusion-resistant fashion.

While it makes analysis easier, this use of randomness can introduce a significant performance penalty. Random data is hard to compress, so storing the database of watermarks requires large memory. In order to decode the watermark found in a pirated copy, one must find a watermark in the database which is similar according to some measure of similarity. Without some structure on the data, this is an expensive operation. Of course, one can impose structure on the watermarks to make storing and decoding more efficient. Our aim here is to do so without losing the security properties obtained for randomly distributed watermarks.

Results: In this paper, we investigate this tradeoff between efficient encodings (encoding many bits per document) and efficient decoding (fast, small memory watermark detection) in two ways. We will use n to denote the length of the data to be marked, m to denote the number of users to be encoded, c to denote the number of colluding pirates, and say that a scheme is c-secure with error ϵ if it has error probability at most ϵ against up to c randomly chosen colluding users.

First, we modify the CKLS watermarking scheme to allow for efficient watermark decoding, and show that this new scheme is secure using the the media watermarking model presented in [6]. The authors of that paper analyze the CKLS scheme, and, in the context of this model, show that the CKLS scheme is c-secure if the document has length $n = \Omega(c^2 \log m)$. Subsequently, this has been shown to be optimal: any scheme in this model requires n of this size [4]. However, the algorithm used for watermarking detection in the CKLS scheme is quite inefficient, requiring time and memory which are linear in the number of

users. We present a modified scheme whose running time depends only *polylogarithmically* on the number of users, at the expense of increasing the dependence of the document length on c.

Theorem 1. *Let* $\gamma = \log c + \log 1/\epsilon + \log\log m$. *The Modified-CKLS scheme can encode* m *users into* $n = O(\gamma c^4 \log m)$ *coordinates in a* c*-secure fashion with error probability at most* ϵ. *The decoder requires memory* $O(\gamma c^6 \log m)$ *and runs in time polynomial in* $c, \log m,$ *and* $\log 1/\epsilon$.

This modified scheme makes use of a two-level coding scheme as used in [2] and [1]. In this approach, an inner watermarking code is combined with an outer error-correcting code with very high minimum distance to obtain a code which maintains collusion security. In our setting, there are two main differences from this previous work. First, because the CKLS watermarking code is very efficient in terms of the number of users, we can make use of constructive codes without requiring n to be too large. In particular, we will be able to use Reed-Solomon codes in a parameter range where the algorithms of [5] provide efficient decoding. Second, previous analyses of algorithms using this two-level approach considered models in which errors in the inner watermarking codes were completely independent. In the media watermarking model, we are guaranteed that a suspect document is not too distorted, but only in an overall sense. Small parts of the documents could still experience very large distortion, possibly preventing decoding of some inner codes.

Second, we present some formal evidence for this intuition that randomness is needed to obtain collusion-resistance. We present a model for *invertible* watermarking schemes. This framework includes *additive* schemes like CKLS which add a document-independent watermark to the document to produce the marked document. We study the case where the scheme is public except for a small amount of secret information known to the encoder and decoder. In this framework, we obtain the following result:

Theorem 2. *Any* c*-secure public invertible watermarking scheme with error at most* ϵ *must have at least* $\Omega((1-\sqrt{\epsilon})c\log\epsilon)$ *bits of secret information.*

In particular, if ϵ is a constant, this implies that there must be at least $\Omega(c)$ bits of secret information.

Organization: In Section 2, we begin with the media watermarking model from [6] and the CKLS scheme. These are important components of our watermarking scheme and its analysis, which we present in Section 3. In Section 4, we define public, invertible watermarking schemes with secret information and then prove Theorem 2, which lower bounds the size of the secret information for such schemes.

2 Media Watermarking

In order to explain the modified CKLS algorithm and its proof of security, it is necessary to first explain the original algorithm and the model used in its anal-

ysis. We begin with some notation for dealing with normal (Gaussian) random variables, which will be useful in describing our model and algorithms.

Definition 1. *$N(0,\beta)$ denotes a normal variable with variance β. $\boldsymbol{N}_n(0,\beta)$ denotes a vector of length n whose components are drawn independently according to $N(0,\beta)$.*

The model we use in analyzing watermarking schemes is the statistical model developed in [6]. This model consists of three main assumptions:

- Documents are real vectors of length n. The original document $\boldsymbol{V}$ is drawn from the distribution $\boldsymbol{N}_n(0,1)$.
- A copy $\boldsymbol{V'}$ of $\boldsymbol{V}$ is *valid* iff $||\boldsymbol{V'} - \boldsymbol{V}|| < \delta\sqrt{n}$ for some $0 < \delta < 1$. All documents produced by either the document owner or the adversary must be valid.
- Besides the information described above, the adversary has only the information present in the copies he has access to. The adversary does not have access to a watermark detector, makes one forged copy, and is exposed on failure.

Assumptions like these are needed in the analysis of any watermarking scheme. To see this, consider two trivial attacks: In the first attack, the adversary knows what the original data is (from prior knowledge or through other sources) without relying on his copies. If an adversary knows what the original document is, then watermarking is futile; the first assumption quantifies the uncertainty that an adversary has about the original document. Similarly, an adversary can redistribute grossly distorted versions of the data that leak no information about his copies. However, significantly distorting the document, either by the adversary or the document owner, renders the document useless. The second assumption rules out attacks of this latter kind. Essentially, it says that documents are represented in such a way that the representation is meaning-preserving: perceptual distance in documents and Euclidean distance of their representations are closely related. Low-level registration attacks, such as StirMark [8, 7], succeed by attacking schemes where this is not true, making small perceptual changes which have large effects on the representation used by that scheme.

Throughout, n will refer to the lengths of the original document and all modified copies as vectors. To indicate that a document is valid with respect to a specific value of δ, we say that it is *δ-valid*.

This model is then used to analyze the CKLS scheme. Since we will use the CKLS scheme as a subroutine, we sketch the encoding and decoding algorithm below.

The encoding algorithm has a strength parameter α, which is chosen to be some constant slightly less than δ.

CKLS-Encoding (original document $\boldsymbol{V}$, user i)
  For $1 \leq i \leq m$
    Draw $\boldsymbol{X^i}$ from $\boldsymbol{N}_n(0,\alpha^2)$

Return marked copy $\boldsymbol{Y^i} = \boldsymbol{V} + \boldsymbol{X^i}$

The decoder makes use of a threshold parameter t.

CKLS-Decoding (suspect document $\boldsymbol{V'}$, original document $\boldsymbol{V}$)
$\boldsymbol{X'} = \boldsymbol{V'} - \boldsymbol{V}$
For each $1 \leq i \leq m$
If $S(\boldsymbol{X'}, \boldsymbol{X^i}) > t$, return i /* i is guilty */

where the similarity measure $S(\boldsymbol{x'}, \boldsymbol{x}) = \boldsymbol{x'} \cdot \boldsymbol{x}/||\boldsymbol{x'}||$.

The choice of t is then made to balance the tradeoff between incriminating innocent users (false positive errors) and allowing guilty users to evade detection (false negative errors). The main lemma from [6] proves the collusion security of this scheme for an appropriate choice of t.

Lemma 1. *[6] Let*

$$n \geq (c + O(1))^2 G \ln(m/\sqrt{p})$$

where $G = \frac{\alpha}{2}\left(1/\delta - 1 - O(n^{-1/12})\right)$. *Then the CKLS scheme*

- *has false positive probability at most* p
- *has false negative probability at most* $O(2^{-n^{1/3}})$, *if there are at most* c *colluders and the attacked document is required to be* δ*-valid.*

We restate it in this form to emphasize a property which will be important in our analysis: The false positive probability is independent of the validity of the attacked document.

3 Modified Scheme

Now, we present our modified CKLS scheme. As in the original, our goal will be to produce a watermarking scheme which encodes the identities of m users into a document of length n. Furthermore, the scheme should be c-secure with error ϵ given the assumptions of the model. Our focus here will be on improving the running time and memory requirements of the decoding algorithm.

At a high level, our scheme works by dividing the n coordinates into s groups, and applying a CKLS scheme within each group. First, we define a few parameters: Let $\delta' = (2\delta + 1)/3$, $r = 1 - (2\delta/(\delta + 1))^2$, and $s = \frac{2c^2}{r^2} \log m$. Given our set of n coordinates, divide them into s groups $G_1, \ldots, G_s$, each consisting of n/s consecutive positions. Given a vector $\boldsymbol{Z}$ of length n, let $\boldsymbol{Z_{(j)}}$ be the vector of length n/s obtained by restricting $\boldsymbol{Z}$ to the positions in G_j.

Watermarking code:

We will make use of s inner codes, $\mathcal{W}_1, \ldots \mathcal{W}_s$, each of which is an independently generated CKLS code with the following properties:

- It encodes the identities of s users.

- The document length is $\ell = O(c^2 \log(s/\epsilon))$.
- All marked copies are δ-valid.
- The probability of false positives is at most ϵ/s.
- If we require that all documents are δ'-valid, and there are at most c colluders, the probability of false negatives is $o(1)$.

If we choose $\alpha < \delta$ and apply Lemma 1, such codes exist and can be generated randomly; furthermore, each watermark can be tested for δ-validity as it is created. These parameters enable us to prevent false positives (which could have unfortunate effects on decoding our outer code) while avoiding the unnecessary expense of preventing false negatives. Let $\boldsymbol{A_{i,j}}$ be the vector of length ℓ which is the mark associated with user i in $\mathcal{W}_j$. Let $\mathcal{D}_j^W(V', V)$ be the watermark detection algorithm associated with the code $\mathcal{W}_j$, as sketched in Section 2, which returns the number (between 1 and s) of an incriminated user. To simplify matters, when decoding a watermark code, we assume that at most one implicated user is returned (if more than one is, we choose one arbitrarily). If no user is implicated, the watermark detector indicates this by returning the special symbol $\emptyset$ indicating an erasure.

Error correcting code:

These inner codes will be combined using an outer, error-correcting code. An $[N, K, D]_q$ *error correcting code* $\mathcal{C}$ over an alphabet $\Sigma, |\Sigma| = q$ is a subset of Σ^N with $|\mathcal{C}| = q^K$. Furthermore, this set has the property that for any $w_1 \neq w_2 \in \mathcal{C}$, the Hamming distance $d(w_1, w_2) \geq D$. For convenience, we will assume that $\Sigma = [1, \ldots, q]$ throughout.

Here, we will use a Reed-Solomon code with length N (and thus field size q) equal to s, dimension $K = \log m$, and distance $D = N - K + 1$. Let w_i be the codeword returned by the encoder $\mathcal{E}$ on input i, and let $\mathcal{C} = \{w_1, \ldots, w_{|\mathcal{C}|}\}$ be the set of codewords. We will examine the decoding process in more detail later. For now, let the decoder $\mathcal{D}(y)$ be an algorithm which, given a word y, returns some codeword $x \in \mathcal{C}$ for which $d(x, y)$ is minimal. The symbol $\emptyset$ in a codeword is interpreted as an erasure error.

3.1 Scheme

Encoding Algorithm (original document $\boldsymbol{V}$, user i)
  For each $1 \leq b \leq s$,
    Let $a \in \Sigma$ be the symbol at position b in w_i
    Let $\boldsymbol{X^i_{(b)}} = \boldsymbol{A_{a,b}}$
  Let $\boldsymbol{X^i}$ be the concatenation $\boldsymbol{X^i_{(1)}} \cdots \boldsymbol{X^i_{(s)}}$.
  Return $\boldsymbol{Y^i} = \boldsymbol{V} + \boldsymbol{X^i}$

Decoding Algorithm (suspect document $\boldsymbol{V'}$, original document $\boldsymbol{V}$)
  For each $1 \leq j \leq s$
    $B_j = \mathcal{D}_j^W(\boldsymbol{V'_{(j)}}, \boldsymbol{V_{(j)}})$ /* Decode the jth inner code */
  Let B be the concatenation $B_1 \cdots B_s$

Let $w_i = \mathcal{D}(B)$ /* Decode the error-correcting code */
If $d(w_i, B) \leq (1 - r/c)s$, incriminate user i

3.2 Analysis

To prove the security of the scheme, we must show three things: all watermarked documents are valid, false positive errors are unlikely, and false negative errors are unlikely. The first is easy: all watermarked documents are trivially δ-valid, since each watermark $\boldsymbol{X^i}$ is the concatenation of a collection of shorter vectors, each of which is itself δ-valid.

False Positives: By our choice of parameters, each inner code has at most an ϵ/s probability of a false positive, so the probability that there is a false positive decoding of any inner code is at most ϵ. Furthermore, this remains true regardless of the distortion with respect to that inner code.

Given a word $w \in (\Sigma \cup \emptyset)^s$, we say that a coalition covers a coordinate if, for some user i in the coalition of the coalition, w_i matches w on that coordinate. We say that w is covered by the coalition if every coordinate where $w \neq \emptyset$ is covered by the coalition. Unless a false positive error occurs, the only codewords which can be produced by a coalition are those words that it covers.

By our choice of parameters, the number of coordinates in which two distinct codewords overlap is at most $\log m + 1$. Thus, for any innocent user, a coalition of c users covers at most $c(\log m + 1) < 2c \log m < rs/c$ coordinates of any codeword belonging to that user, and the coalition is incapable of incriminating him.

False Negatives: Unlike the case of the false positives, if an inner code experiences too much distortion, it is quite likely to produce a false negative error; therefore, we cannot hope to simply drive down the probability of these errors by our choice of parameters. However, not too many groups can be so distorted without rendering the document invalid.

For each group G_i, let its weight $h_i = ||\boldsymbol{V'_{(i)}} - \boldsymbol{V_{(i)}}||^2$. Call a group G_i heavy if $h_i > (\delta')^2 \ell$. Note that the number of heavy groups is at most $(\delta/\delta')^2 s$, since otherwise $||\boldsymbol{V'} - \boldsymbol{V}||^2 = \sum_i h_i \geq \delta^2 n$ and $\boldsymbol{V'}$ is invalid.

For each non-heavy group, incorrect decoding of the inner CKLS code happens with probability only ϵ/s, so with probability $1-\epsilon$, all non-heavy groups are decoded correctly. For the heavy groups, we are not guaranteed correct decoding of the corresponding inner code because δ'-validity is violated. Assuming that no inner code decoding produced an error, the number of empty coordinates (due to either heavy groups or false negatives from the inner code) is less than $(1-r)s$. Thus, at least one member of the coalition agrees with the decoded word w' on rs/c symbols and is implicated.

3.3 Efficiency

The key measures of efficiency are the document length n needed to encode m users in a c-secure fashion, the storage required by the decoder, and the running time of the decoding algorithm. Let $\gamma = \log c + \log 1/\epsilon + \log\log m$.

Document Length: The length of each inner code is $O(c^2 \log s/\epsilon)$. The length of the outer code is $s = \frac{2c^2}{r^2} \log m$. Since r is a constant depending only on the constant δ, the total document length is then $O(\gamma c^4 \log m)$.

Memory: There are s inner codes with s users each, and each such user requires storage of a vector of length $O(c^2 \log s/\epsilon)$, for a total of $O(s^2 c^2 \log s/\epsilon) = O(\gamma c^6 \log^2 m)$.

To obtain efficient decoding, we will make use of the recent list-decoding algorithms for Reed-Solomon codes due to [5]. These algorithms output a list of all codewords which are sufficiently close to the input word in polynomial time, assuming some conditions are met. Given such a list, it is easy to extract a codeword minimizing this distance.

Lemma 2. *[5] The list-decoding problem for $[N, K+1, D]_q$ Reed-Solomon codes allowing for e_1 errors and e_2 erasures can be solved in polynomial time, provided $e_1 + e_2 < N - \sqrt{(N-e_2)K}$.*

We state the result in this form only to emphasize the fact that these decoding methods handle erasures directly, rather than by turning them into errors, which makes our lives simpler.

This requirement can easily be restated to say that the number of non-error coordinates is at least $\sqrt{NK}$. Since we want to detect whether or not there is a word with at least $rs/c = (2c \log m)/r$ non-errors, and $\sqrt{NK} = (\sqrt{2}c \log m)$, the decoding algorithm of [5] succeeds. The running time is polynomial in the document length, which is polynomial in $c, \log m$, and $\log 1/\epsilon$.

Summarizing the results of this section:

Theorem 1. *Let $\gamma = \log c + \log 1/\epsilon + \log\log m$. The Modified-CKLS scheme can encode m users into $n = O(\gamma c^4 \log m)$ coordinates in a c-secure fashion with error probability at most ϵ. The decoder requires memory $O(\gamma c^6 \log^2 m)$ and runs in time polynomial in $c, \log m$, and $\log 1/\epsilon$.*

4 Lower Bounds

To get some insight into the requirements that collusion-security places on a watermark detector, we examine the amount of secret memory a detector must have in a public invertible watermarking scheme. We show that as an adversary sees more documents, he gains some insight into whatever secret information the scheme uses. If the scheme does not have enough secret information compared to the coalition size, the adversary has an attack which defeats the watermarking scheme. To make this intuition formal, we must first define what we mean by public and invertible.

- The encoder is a function which computes a marked document for user i from the user ID i, the original document, and a secret input of b bits common to all users.
- The decoder is a function of the suspect document and the secret input. It has the property that the document given to user i incriminates user i.
- Both algorithms are known to the users. Also, the user ID is public (ie, user i knows that he is user i).
- Given the value of the secret bits, a user, and a marked document, there is an algorithm which returns the corresponding original document.

The first two conditions simply describe the necessary framework, the third defines a public scheme, and the last defines invertibility. Many natural schemes are captured in this notion of invertibility, such as *additive* schemes like CKLS and replacement schemes like the Boneh-Shaw algorithm. (The model used to analyze the Boneh-Shaw algorithm, however, is incompatible with this model because it limits the changes which can be made by an adversary by assuming that some changes made by the encoder are not detectable by the users). Invertible schemes have been studied previously because of the issues they present in proving ownership; here, however, we focus on their security. This assumption gives an adversary some hope of defeating a watermarking scheme by removing his watermark and replacing it by another. Finally, it is not hard to envision schemes which somehow evade these restrictions. A scheme which made use of pseudorandomness could potentially have access to many free "random" bits whose effect is not captured by these information-theoretic definitions. Similarly, schemes which compute the mark as a function of the original document may not be captured by this model, as inverting the watermarking process without the original document may be difficult. New analyses of watermarking schemes which make significant use of pseudorandomness or non-invertibility in a collusion-secure way would be very interesting.

Our aim now is to show that in any public, invertible scheme with small error probability, b, the number of secret bits, must be sufficiently large. We begin with a key lemma.

Lemma 3. *If $b < (1-\delta)c\log\delta$, for any set of $c+1$ users $U = \{u_1, \ldots, u_{c+1}\}$, there are at least δc users $u_i \in U$ such that u_i can be incriminated by $U - u_i$ with probability at least δ.*

Proof. Since the algorithm is public except for the secret information, our aim will be to pin down this secret information by comparing documents. The axioms of the model will allow us to perform the following key operation: Given the documents d_i, d_j of two users u_i, u_j, it is possible to tell if a given value of secret information b is consistent with this view: Using this value of b, invert d_i for user i to obtain V, and then re-encode V for user j and see if it equals d_j. By checking pairwise relations, we can determine if any set of at least 2 documents are consistent with a value b.

Consider the following process: Go through the sets $\{u_1\}$, $\{u_1, u_2\}$, ..., $\{u_1, \ldots, u_{c+1}\}$ in order. At each stage i, we will keep track of a set B_i of possible

values of b which are consistent with the documents belonging to the first i users. Initially, $|B_1| = 2^b$. Call a user j *directly determined* if, given $\{u_1, \ldots, u_{j-1}\}$, there is some document d'_j which is consistent with all previous d_i and an δ-fraction of the remaining values B_{j-1}. For each j, note that if user j is not directly determined, then $|B_j| \leq \delta |B_{j-1}|$, since whatever the value of d_j is, it is inconsistent with many values of b. If user j is directly determined, then the first $j-1$ users (and therefore $U - u_j$) can incriminate j with probability at least δ. To do so, they choose a value $b \in B_{j-1}$ randomly, invert one of the documents with respect to b, and then re-encode this inverted document for user j. If the size of B ever reaches 1, then the coalition knows the secret information, and is thus capable of removing the mark or incriminating any user. In this case, all remaining users in U are declared *indirectly determined*.

There are at least $c - b/\log\frac{1}{\delta}$ determined users, since there are at most $b/\log\frac{1}{\delta}$ times when the next user is not determined before all remaining users are indirectly determined, and the lemma follows.

Theorem 2. *Any c-secure public invertible watermarking scheme with error at most ϵ must have at least $\Omega((1-\sqrt{\epsilon})c\log\epsilon)$ bits of secret information.*

Proof. We say that a set of users can α-*implicate* another user if they have some strategy which succeeds in doing so with probability at least α.

We begin by applying the previous lemma with $\delta = 2\sqrt{\epsilon}$. From this, for each set of $c+1$ users, we can obtain δc rules of the form "$u_1, \ldots, u_c$ can δ-incriminate u_{c+1}." In total, there are $\delta c\binom{m}{c+1}$ such rules. Each set of c users can implicate at most $m - c$ users, so there are at least $\delta c\frac{1}{m-c}\binom{m}{c+1}$ sets of c users which δ-implicate some other user.

Since there are $\binom{m}{c}$ coalitions of size c, the probability that the adversary gets a coalition which can δ-implicate another user is at least

$$\delta c\frac{1}{m-c}\binom{m}{c+1}\Big/\binom{m}{c} = \delta\frac{c}{c+1} \geq \frac{\delta}{2} = \sqrt{\epsilon}$$

Given such a coalition, he succeeds in implicating an innocent user with probability at least $\delta > \sqrt{\epsilon}$, for an overall success probability of at least ϵ.

5 Conclusion

We presented improvements to the CKLS watermarking scheme allowing for efficient watermark decoding by reducing randomness using error-correcting codes, and demonstrated that for a natural class of watermarking schemes, some randomness is necessary to obtain collusion resistance.

The same technique of applying a two-level coding scheme using an efficiently decodable error-correcting code can also be applied to the Boneh-Shaw watermarking scheme. Unfortunately, applying it directly leads to disappointing performance. If the outer error-correcting code is a Reed-Solomon code, the field size, and thus the number of users in each inner watermarking code, will need to

be $\Omega(c^2 \log m)$ to obtain the needed $(1 - 1/c^2)$ relative minimum distance (this dependence on $\log m$ can be removed by using algebraic-geometry codes, but at the price of an even worse dependence on c). The inner watermarking code used in [1] encoding m users has length $\Theta(m^3 \log m)$, rather than $\Theta(c^2 \log m)$ in the case of CKLS, so if the alphabet size of the outer code is large ($O(c^2 \log m)$), the length of the resulting code is very long ($O(c^6 \log^3 m)$).

Another approach, suggested in discussions with Robert Tarjan, is to use the entire two-level Boneh-Shaw scheme, including the random error-correcting code, as an inner code. As before, the outer Reed-Solomon error-correcting code has $O(c^2 \log m)$ symbols over an alphabet of the same size. Now, however, this is applied to an inner code which encodes M users c-securely in $O(c^4 \log M)$ bits, for a total length of $O(c^6 \log m \log c \log \log m)$. Now, we are left with the problem of decoding these mid-level random error-correcting codes. The key observation is that since each such code has only $c^2 \log m$ codewords, it can be brute-force decoded in time which is polynomial in both c and $\log m$, as desired.

There are several directions where this work could be extended. First, the issue of watermark detector memory could be applied in other situations; in settings like the Boneh-Shaw model where proving tight lower bounds seems difficult, perhaps they may offer another means of analyzing the complexity of the problem. More ambitiously, this two-level coding scheme gets down to logarithmic complexity (in m) immediately, but leads to an inherent blow-up in the dependence on c which is only made worse by the absence of constructive optimal low-rate codes. In many practical settings, other tradeoff points (for example, complexity $\sqrt{m}$ rather than $\log m$) would be of great interest if the dependence on c could be weakened.

Finally, we would like to thank Robert Tarjan for many helpful discussions regarding watermark decoding and Amin Shokrollahi for answering numerous coding questions.

References

1. D. Boneh and J. Shaw. Collusion secure fingerprinting for digital data. *IEEE Transactions on Information Theory*, 44(5):1897–1905, 1998.
2. B. Chor, A. Fiat, and M. Naor. Tracing traitors. In *Crypto '94*, pages 257–270, 1994.
3. I. Cox, J. Kilian, T. Leighton, and T. Shamoon. Secure spread spectrum watermarking for multimedia. *IEEE Transactions on Image Processing*, 6:1673–1687, 1997.
4. F. Ergun, J. Kilian, and R. Kumar. A note on the limits of collusion-resistant watermarks. In *Eurocrypt '99*, pages 140–149, 1999.
5. V. Guruswami and M. Sudan. Improved decoding of reed-solomon and algebraic-geometric codes. *IEEE Transactions on Information Theory*, 45(6):1757–1767, September 1999.
6. J. Kilian, F. T. Leighton, L. R. Matheson, T. G. Shamoon, R. E. Tarjan, and F. Zane. Resistance of digital fingerprints to collusional attacks. In *Proceedings of 1998 IEEE International Symposium on Information Theory*, page 271, Cambridge, MA, August 1998. Full version available as Princeton CS TR-585-98.

7. F.A.P. Petitcolas and R.J. Anderson. Evaluation of copyright marking systems. In *IEEE Multimedia Systems (ICMCS'99)*, pages 574–579, 1999.
8. F.A.P. Petitcolas, R.J. Anderson, and M.G. Kuhn. Attacks on copyright marking systems. In *Second International Workshop on Information Hiding*, pages 219–239, 1998.