A High-Rate Fingerprinting Code

Jean-François Jourdas\textsuperscript{a} and Pierre Moulin\textsuperscript{a,b}

\textsuperscript{a}Beckman Institute and ECE Department, \\
\textsuperscript{b}Coordinated Science Laboratory, \\
University of Illinois at Urbana-Champaign, \\
405 N. Matthews Ave., Urbana, IL, USA.

\section*{ABSTRACT}

In fingerprinting, a signature, unique to each user, is embedded in each distributed copy of a multimedia content, in order to identify potential illegal redistributors. As an alternative to the vast majority of fingerprinting codes built upon error-correcting codes with a high minimum distance, we propose the construction of a random-like fingerprinting code, intended to operate at rates close to fingerprinting capacity. For such codes, the notion of minimum distance has little relevance. As an example, we present results for a length 288,000 code that can accommodate 33 millions of users and 50 colluders against the averaging attack. The encoding is done by interleaving the users’ identifying bitstrings and encoding them multiple times with recursive systematic convolutional codes. The decoding is done in two stages. The first outputs a small set of possible colluders using a bank of list Viterbi decoders. The second stage prunes this set using correlation decoding. We study this scheme and assess its performance through Monte-Carlo simulations. The results show that at rates ranging from 30\% to 50\% of capacity, we still have a low error probability (e.g. 1\%).

\textbf{Keywords:} Fingerprinting Code

\section{1. INTRODUCTION}

Fingerprinting aims at protecting digital contents such as music, images and videos from illegal redistribution. A signature unique to each user, called the user’s fingerprint, is embedded into a content without perceptually modifying the host. A group of users can collude to produce a pirated version of the content in which their fingerprints are attenuated but the host itself is unchanged. Two-cost effective collisions are the averaging collusion, in which the users’ contents are averaged, and the interleaving collusion, in which each user contributes different parts of his content to produce a new one. After colluding, the pirates add a low level noise to try to wash out their fingerprints without degrading too much the host.

Existing designs of fingerprinting codes based on error-correcting codes rely on the notion of high minimum distance. This is motivated by algebraic constructions for traceability codes, e.g. Silverberg \textit{et al.}\textsuperscript{1} and Barg \textit{et al.}\textsuperscript{2} who established the mathematical properties of algebraic and concatenated codes with high minimum distance. In subsequent works, other authors have used minimum distance separable codes. Fernandez \textit{et al.}\textsuperscript{3} proposed a soft-decision tracing fingerprinting scheme, under the Boneh-Shaw marking assumption,\textsuperscript{4} based on Reed-Solomon coding and the Guruswami-Sudan\textsuperscript{5} decoding algorithm. He and Wu\textsuperscript{6,7} proposed a joint coding-embedding scheme based on Reed-Solomon codes. The first design\textsuperscript{6} assumes a small number of users (1,024) in which case the decoding is done by brute force search to identify the user whose fingerprint has the highest correlation with the forgery. In the second design,\textsuperscript{7} they used trimming symbols to reduce the complexity of the correlation detector.

Another interesting design is the Anti-Collusion Codes (ACC) introduced by Trappe \textit{et al.}\textsuperscript{8}. The main property of an ACC is that under the Boneh-Shaw marking assumption,\textsuperscript{4} the composition of any subset of $K$ or

Further author information: (Send correspondence to J-F Jourdas) 
J-F Jourdas: E-mail: jourdas2@uiuc.edu, Telephone: 1 217 819 9312 
P. Moulin: E-mail: moulin@ifp.uiuc.edu, Telephone: 1 217 244 8366
less fingerprints is unique. Using this property, it is possible to trace all the colluders. These codes are easy to implement but no performance guarantees were given in terms of minimum distance or resilience to noise.

To model the collusion attack, we use the notion of signal distortion instead of the Boneh-Shaw marking assumption because the notion of signal distortion is central to media fingerprinting.

Some fundamental performance limits for fingerprinting have been reported in recent literature. The rate of a fingerprinting code of length $N$, accommodating $M$ users, is defined as $R = \frac{1}{N} \log_2 M$. When $R$ exceeds the capacity $C$ of the collusion channel, it becomes impossible to identify reliably the colluders. Conversely, when $R < C$ and $N$ is large enough, there exists a code with vanishing error probability.

Y. Wang derived an expression for $C$ under the assumption of a Gaussian host and mean-squared distortion constraints on the embedder and the colluders (see for the proof when $K = 2$):

$$C = \frac{1}{2K} \log_2 \left(1 + \frac{D_t}{D_c} \frac{K}{2} \right),$$

where $D_t$ is the maximum distortion per sample between the original host and the fingerprinted version, $D_c$ is the maximum distortion per sample that the colluders can introduce, and $K$ is the number of colluders. For example, when $K = 10$ and $D_t = D_c = 1$, we obtain $C = 0.05$. The capacity expression holds for both blind and nonblind fingerprinting. For nonblind fingerprinting, capacity is achieved using additive embedding: $x^k = s + \sqrt{D_t} t^k$, where $t^k$ is the unit-power fingerprint assigned to user $k$, and $s$ is the host signal.

In many fingerprinting problems, the rate is much lower than capacity, and fingerprinting codes with high minimum distance are desirable. This paper focuses on the opposite regime where the rate is close to capacity. Classical information theory tells us that it is the distance distribution of the code and not the minimum distance that matters in this high-rate regime; moreover, random-like codes perform optimally in this regime. The most famous high-rate codes for point-to-point communications are turbo-codes and they have poor minimum distance.

In practice, we would like to build a code with rate $R$ close to the capacity $C$, i.e., very short, withstanding large coalitions and able to detect at least one of the colluders with error probability $P_e$ of the order of $10^{-2}$ to $10^{-3}$.

2. A HIGH-RATE CODE BASED ON TURBO-CODE STRUCTURE

The encoding scheme is diagrammed in Fig. 1. A user is identified by a length-$n$ bitstring $u$. The number of users that can be accommodated is $M = 2^n$. First, the bitstring $u$ is encoded with a Recursive Systematic Convolutional (RSC) code of rate $R_1$ into a binary codeword which is then mapped to a binary antipodal...
sequence $c_1$ of length $\sim n/R_1$ (the approximation is only moderately good due to the use of tail bits in the RSC code). We will refer to $c_1$ as a subcodeword. Following this step, $u$ is interleaved and encoded with the same or another RSC code of rate $R_2$ to get a different subcodeword $c_2$ of length $\sim n/R_2$. This last operation is repeated a total of $N_i$ times. Then, the subcodewords $c_i$, $1 \leq i \leq N_i + 1$, generated are concatenated to form a sequence $c$ of length $N \sim n \sum_{i=1}^{N_i+1} 1/R_i$. As depicted in Fig. 2, we map $c$ to a sequence in $\mathbb{R}^N$ in order to embed the fingerprint in a real-valued host sequence. For all $1 \leq i \leq N_i + 1$, we apply the Discrete Cosine Transform (DCT) to each block of length $1/R_i$ that makes up the subcodeword $c_i$ and obtain a real sequence $f_i$. Then, the sequences $f_i$, $1 \leq i \leq N_i + 1$, are concatenated to form a real fingerprint $f$. We chose the DCT because it is a global transform which is easy to implement and spreads the energy over all the coefficients. Nevertheless, any other unitary transform could be used, and different transforms could even be used in different blocks.

![Figure 2. Orthornormal transforms mapping a fingerprint in \{-1,1\}^N to a unit-power fingerprint in \mathbb{R}^N.](image)

This code does not have a high minimum distance but the presence of interleavers makes the $N_i + 1$ codewords uncorrelated, thus we can gain by exchanging information after decoding each of them.

The code has many parameters that determine its performance. First, we must choose the polynomial generators $\{g_i\}$ of the RSC codes and their rates $\{R_i\}$. There might be optimal choices for the $\{g_i\}$ depending on the collusion attack expected but we have not explored this issue yet. However, we noticed during our simulations that randomly choosing the $\{g_i\}$ with many 1’s in the generator polynomial matrices representing them yields good performance. For our simulations, we generated the components of the generator polynomial matrices independently according to a Bernoulli law with parameter 0.9.

Second, for a given length $N$ of the fingerprint, there is a tradeoff between $N_i$ and the values of the different rates $\{R_i\}$. For simplicity, we took all the rates $\{R_i\}$ identical and we will refer to their common value as $R_1$. If we increase $N_i$, we must also increase $R_1$ to preserve the length $N$ and vice-versa. When $R_1$ is low, each segment $f_i$, $1 \leq i \leq N_i + 1$, is more robust to collusion attacks because the different codewords of the $i$th RSC code are more apart from each others. However, as in this case, $N_i$ is low too, there are fewer interleavers and less randomness in the code. On the contrary, when $N_i$ is high, the code becomes very randomized. However, as $R_1$ is high too, each segment $f_i$, $1 \leq i \leq N_i + 1$, is less resilient to collusions. Simulations made with the decoding algorithm of Sec. 3 suggest that it is more beneficial to have a rather low value for $N_i$ and a low rate $R_1$. In our simulations, we usually set $N_i = 3$ and then modify $R_1$ to change the length $N$. Note that it is not a good idea to have $N_i$ lower than 3 because the concatenation of the user’s identity encoded differently is what makes the code random-like.

Third, we must choose the different interleavers. There might be an optimal choice for the sets of interleavers but we have not studied this question yet. Although it seems that random interleavers are the better ones.

### 3. DECODING SCHEME

The decoder receives a real length-$N$ sequence $r$, called forgery, which is the output of the collusion channel, whose inputs are the colluders fingerprints (see Fig. 3). For instance, the collusion channel could perform an interleaving or averaging followed by addition of noise.

The forgery $r$ can be broken into $N_i + 1$ consecutive subsequences $\{r_i\}$ of the same length. Each subsequence $r_i$, $1 \leq i \leq N_i + 1$, is decoded by a list Viterbi algorithm, which outputs an ordered list $L_i$ of the $S$ most likely
colluders to have participated in its creation. As the input to the Viterbi algorithm is a real-valued sequence, the metric is Euclidean distance. Our simulations point out that, usually, one list contains one or many colluders but they are not at the top of the list. However, by comparing the users accused in different lists, we gain reliability.

A possible way to identify only one of the colluders is to accuse, among the users in the union $L_f$ of all lists $\{L_i\}$, the most likely one to have taken part in the creation of the forgery. That is to say, among all the users in the final list $L_f$, we do an exhaustive search to select the one whose fingerprint has the highest correlation with the forgery $r$. This decoding algorithm is summarized in Fig. 4.

The only parameter of the decoding algorithm is the value of $S$. The bigger $S$, the bigger the list $L_f$ and thus the better the detection of one colluder. However, the computational time increases. For our simulations, we usually set $S = 512$, thus when $N_i = 3$, the size of the list $L_f$ is 2,048. If we choose $n = 25$, there are 33 millions

![Figure 3. Collusion attack carried out on the code when the fingerprint is in $\mathbb{R}^N$. In this paper, we consider the averaging attack followed by additive white Gaussian noise.](image)

![Figure 4. Description of the decoding scheme.](image)
of users and the matched filter detection is only done on a tiny subset of the original set of users.

From the decoding point of view, we can also analyze the tradeoff between \( N_i \) and the values of the different rates \( \{ R_i \} \) for a given fingerprint length \( N \). The number of users in \( L_i \) is \((N_i + 1)/S\). A high value of \( N_i \) means a large number of users in \( L_i \). But, as \( R_1 \) is high too, the lists \( \{ L_i \} \) are less reliable because the RSC codes cannot sustain large collusions. In other words, we have more users in \( L_i \) on which to apply the matched filter detection but it is less likely that \( L_i \) contains colluders. For instance, in the case \( L_i \) contains none, the matched filter has no chance of identifying correctly a colluder. Simulations point out that it more important to have reliable lists \( L_i \) even though there are fewer users to choose from in the correlation detection.

## 4. PERFORMANCE EVALUATION

The number of users \( M \) and the code length \( N \) are determined by the fingerprinting problem considered. In our simulations, we targeted 30 millions users and fingerprint lengths of 1,000 to 1,000,000. These numbers are typical of image and video fingerprinting for large-scale redistribution. A targeted number of colluders is fixed, then the parameters of the code are chosen through simulations to get the shortest code with low error probability. There are two measures of performance: the probability \( \hat{P}_e \) that the user accused by the algorithm turns out to be innocent and the ratio \( R/C \) that indicates how close the rate is to capacity. Due to the computational complexity of the decoding algorithm, we limited \( S \) to 512 when \( N \) exceeds 1,000. But this limitation is purely practical because we would like to have \( S \) as big as possible to improve the performance.

We studied the averaging attack followed by the addition of white Gaussian noise of variance \( D_c \) (Fig. 3). We assume that \( s \) is available at the decoder and choose \( D_t = D_c = 1 \). \( K \) represents the number of colluders. SNR is defined as \( D_t/D_c \). Thus, here, SNR=1. To estimate the different probabilities of interest, we performed 100 to 2,000 Monte Carlo simulations. The results are displayed in Table 1. We also report in Table 2, the performance of existing fingerprinting codes in the same conditions.

### Table 1. Numerical experiments. The collusion is an averaging attack followed by additive white Gaussian noise. \( P_e \) represents the probability not to catch any colluders. \( N \) is the length of the code, and \( M \) is the number of users. \( R \) is the rate of the code, and \( C \) is the capacity of the collusion channel. SNR=\( D_t/D_c \), and \( K \) is the number of colluders.

<table>
<thead>
<tr>
<th>( N )</th>
<th>( M )</th>
<th>( R )</th>
<th>( C )</th>
<th>( R/C )</th>
<th>( N_i )</th>
<th>( SNR )</th>
<th>( R_1 )</th>
<th>( N_{\text{tests}} )</th>
<th>( S )</th>
<th>( K )</th>
<th>( P_e )</th>
</tr>
</thead>
<tbody>
<tr>
<td>784</td>
<td>33,554,432</td>
<td>0.032</td>
<td>0.069</td>
<td>0.46</td>
<td>3</td>
<td>1</td>
<td>1/7</td>
<td>1,000</td>
<td>1024</td>
<td>3</td>
<td>0.009</td>
</tr>
<tr>
<td>7,440</td>
<td>33,554,432</td>
<td>0.0034</td>
<td>0.0069</td>
<td>0.49</td>
<td>3</td>
<td>1</td>
<td>1/60</td>
<td>2,300</td>
<td>512</td>
<td>10</td>
<td>0.0074</td>
</tr>
<tr>
<td>230,400</td>
<td>33,554,432</td>
<td>0.00011</td>
<td>0.00029</td>
<td>0.38</td>
<td>3</td>
<td>1</td>
<td>1/1600</td>
<td>421</td>
<td>512</td>
<td>50</td>
<td>0.019</td>
</tr>
<tr>
<td>288,000</td>
<td>33,554,432</td>
<td>8.68 \times 10^{-5}</td>
<td>2.86 \times 10^{-4}</td>
<td>0.30</td>
<td>3</td>
<td>1</td>
<td>1/2000</td>
<td>161</td>
<td>512</td>
<td>50</td>
<td>0.012</td>
</tr>
</tbody>
</table>

The main result is that our codes operate at rates ranging from 30% to 50% of capacity, with error probabilities in the range of 0.1% to 1%.

With a code of length of only 7,440, we accommodate 33 millions of users and resist up to 10 colluders against the averaging attack with SNR of 1 and error probability of 0.1%. The rate of this code is 49% of capacity. For comparison, Trappe \textit{et al.}\(^8\) used a modulated ACC of length 10,000 accommodating 20 users and resisting 3 colluders against averaging attack. With a code of comparable length, we can accommodate 1,000,000 times more users and resist to 3 times more colluders. Moreover, the rate of the ACC code\(^8\) is only 7% of capacity.

With a code of length \( N = 288,000 \), we accommodate 33 millions users and resist up to 50 colluders with a SNR of 1 and error probability of 1%. The rate of this code is 30% of capacity. For comparison, He and Wu\(^7\) used a code of length 261 millions accommodating 16 millions of users and resisting up to 100 colluders under the averaging attack with a small probability of error too (as it appears from Fig. 9(a) of \(^7\)). Although the number of users and colluders are of the same order, this code is 1,000 times longer than ours. Indeed, its rate is 0.1% of capacity.

From Table 1, we observe a decrease of the ratio \( R/C \) from the code of length 7,440 to the code of length \( N = 288,000 \). When there are more colluders, the fingerprints are harder to recover and the size of the list \( L_i \)
Table 2. Performance of existing fingerprinting codes against the averaging attack followed by additive white Gaussian noise. For He and Wu’s code,\textsuperscript{7} the attack is also followed by MPEG2 compression. We used the notation $\sim 0$ to mean that the plots given in the articles were not precise enough to read $P_e$.

<table>
<thead>
<tr>
<th>Code</th>
<th>$N$</th>
<th>$M$</th>
<th>$R$</th>
<th>$C$</th>
<th>$R/C$</th>
<th>$K$</th>
<th>SNR</th>
<th>$P_e$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACC\textsuperscript{8}</td>
<td>10,000</td>
<td>20</td>
<td>$4.32 \times 10^{-4}$</td>
<td>0.069</td>
<td>0.070</td>
<td>3</td>
<td>1</td>
<td>$\sim 0$</td>
</tr>
<tr>
<td>GRACE\textsuperscript{6}</td>
<td>30,000</td>
<td>1,024</td>
<td>$3.33 \times 10^{-4}$</td>
<td>0.0018</td>
<td>0.019</td>
<td>20</td>
<td>1</td>
<td>0.01</td>
</tr>
<tr>
<td>He and Wu\textsuperscript{7}</td>
<td>261,734,400</td>
<td>16,000,000</td>
<td>$9.14 \times 10^{-8}$</td>
<td>$7.18 \times 10^{-5}$</td>
<td>0.0013</td>
<td>100</td>
<td>1</td>
<td>$\sim 0$</td>
</tr>
</tbody>
</table>

should be increased accordingly. However, this requires too many computations, which is why we kept $S$ at 512 in the second simulation too. Nevertheless, by using a bigger $S$, we should be able to get closer to capacity.

Interestingly, for a target rate $R$, the code length $N = \frac{1}{R} \log_2 M$ does not depend much on the number of users. In Table 1, the codes of lengths $N = 7,440$ and $N = 288,000$ accommodate the same number of users. Actually, the length depends essentially on the number of colluders that the code resists. Collusion resistance could be increased by using a higher rate for the RSC codes or by increasing $N_t$, but this would imply a longer fingerprint.

In addition to being able to accommodate many users and resist many colluders, our system has reasonable decoding complexity. Specifically, decoding complexity is the sum of the complexities of the Viterbi algorithms used and of the matched filter. For long fingerprints, the complexity of the decoding algorithm is dominated by the matched filter whose complexity is $O((N_t + 1)SN)$. Hence, the complexity is linear in $N$. The longer the code is, the longer it takes to decode a codeword, which is why there are less simulations for the longer codes.

5. REDUCTION OF THE PROBABILITY OF FALSE ALARM

The decoding algorithm always accuses a user: the one with the highest correlation. But it is very important not to accuse an innocent user. We would rather declare a decoding failure than accuse someone wrongly. Analyzing the distributions of the correlations of the fingerprints of the users in the final list $L_f$ with the forgery suggests a simple modification of the decoding algorithm which reduces the probability of accusing wrongly someone.

![Figure 5. Distributions of the correlation statistics for innocent users and colluders of the final list $L_f$. The collusion considered is the averaging attack followed by additive white Gaussian noise. The parameters of the simulation are $N = 7,440$, $M = 33,554,432$, $N_t = 3$, $R_1 = 1/60$, $S = 512$, $K = 10$ and $D_f = D_c = 1$.](image-url)
Table 3. Numerical experiments for the thresholding decoding algorithm. The collusion attack is the averaging attack followed by additive white Gaussian noise. $P_{FP}$ is the probability that someone accused is not a colluder. $P_m$ is the probability of declaring a decoding failure.

<table>
<thead>
<tr>
<th>N</th>
<th>M</th>
<th>R</th>
<th>C</th>
<th>R/C</th>
<th>$N_i$</th>
<th>SNR</th>
<th>$R_1$</th>
<th>$N_{test}$</th>
<th>S</th>
<th>K</th>
<th>$\tau$</th>
<th>$P_{FP}$</th>
<th>$P_m$</th>
</tr>
</thead>
<tbody>
<tr>
<td>7,440</td>
<td>33,554,432</td>
<td>0.0034</td>
<td>0.0069</td>
<td>0.49</td>
<td>3</td>
<td>1</td>
<td>1/60</td>
<td>2,000</td>
<td>512</td>
<td>10</td>
<td>975</td>
<td>0.0016</td>
<td>0.044</td>
</tr>
</tbody>
</table>

We ran 1,000 Monte Carlo simulations with a code of parameters $N = 7,440$, $M = 33,554,432$, $N_i = 3$ and $R_1 = 1/60$. As in the previous section, we assume that the host is available to the decoder. There are $K = 10$ colluders and they use the averaging attack followed by additive white Gaussian noise of variance $D_c$. We choose $D_f = D_c = 1$ and $S = 512$.

For each simulation, we generated $L_f$ and split it in two lists: one containing only innocent users and the other containing only colluders. Then, we computed the correlations of the fingerprints of each user with the forgery. Finally, we computed the histograms of the correlations of innocent users and colluders in $L_f$. They are displayed in Fig. 5.

The two distributions are symmetric and look Gaussian. The correlations of innocent users are centered around 535 whereas the correlations of colluders are centered around 1,052. Hence, a threshold can be used to declare if the user identified by the decoding algorithm is a colluder.

In order to increase the reliability of the accusations, the previous scheme is modified as follows. As before, we find the user in $L_f$ whose fingerprint has the highest correlation with the forgery. If the correlation is above a threshold $\tau$, we accuse this user to be a colluder otherwise we declare a decoding failure. To assess the performance of this new algorithm, we ran 2,000 Monte-Carlo simulations with $\tau = 975$ and the same parameters used to compute the histograms. We measured the probability of miss $P_m$ and the probability of false positives $P_{FP}$. The results are shown in Table 3.

The threshold $\tau$ determines the two probabilities $P_{FP}$ and $P_m$. The bigger $\tau$, the bigger $P_m$ and the smaller $P_{FP}$. Even if the new decoding algorithm misses colluders in some cases, the accusations are more reliable. In the simulation of Table 3, the probability of miss is of about 4% which is reasonable. On the other hand, the estimated $P_{FP}$ is only 0.0016. This is a notable improvement with respect to the error probability $P_e$ of the previous decoding scheme which was 0.0074.

6. CONCLUSION

To conclude, we have proposed the first practical fingerprinting code not based on high minimum distance. This is a high-rate code and is based on the fact that random-like codes are optimal when their rates are close to capacity. Despite being short and accommodating millions of users, our code can resist many colluders. For example, we presented simulations results for a code of rate 30% of capacity, accommodating millions of users, resisting 50 colluders against the averaging attack and with a probability of not catching any colluder of about 1%. We also presented an improved decoding algorithm that yields a very low probability of wrongly accusing someone at the cost of declaring a decoding failure in some cases.

ACKNOWLEDGMENTS

Work supported by Hewlett-Packard Laboratories and the National Science Foundation under grants CCF06-35137 and CCF07-29061.
REFERENCES


