Lower Bounds for Collusion-Secure Fingerprinting

Chris Peikert* abhi shelat* Adam Smith*

MIT Laboratory for Computer Science, Cambridge, MA 02139

Abstract

Collusion-secure fingerprinting codes are an important primitive used by many digital watermarking schemes [1, 10, 9]. Boneh and Shaw [3] define a model for these types of codes and present an explicit construction. Their code has length $O(c^3 \log(1/\epsilon))$ and attains security against coalitions of size c with ϵ error. Boneh and Shaw also present a lower bound of $\Omega(c \log(1/c\epsilon))$ on the length of any collusion-secure code.

We give new lower bounds on the length of collusion-secure codes by analyzing a weighted coin-flipping strategy for the coalition. As an illustration of our methods, we give a simple proof that the Boneh-Shaw construction cannot be asymptotically improved. Next, we prove a general lower bound: no secure code can have length $o(c^2 \log(1/c\epsilon))$, which improves the previous known bound by a factor of c. In particular, we show that any secure code will have length $\Omega(c^2 \log(1/c\epsilon))$ as long as $\log(1/\epsilon) \geq Kk \log c$, where K is a constant and k is the number of columns in the code (in some sense, a measure of the code's complexity). Finally, we describe a general paradigm for constructing fingerprinting codes which encompasses the construction of [3], and show that no secure code that follows this paradigm can have length $o(\frac{c^3}{\log c} \log(1/c\epsilon))$ (again, by showing a lower bound for large values of $\ln(\frac{1}{\epsilon})$). This suggests that any attempts at improvement should be directed toward techniques that lie outside our paradigm.

1 Introduction

Watermarking has been an effective security tool for centuries. For example, in the French Decorations scandal of 1887, a paper watermark established that two political letters supposedly written in 1884 were actually written on paper manufactured in 1885. These antedated letters created a political scandal which forced the prime minister to resign. In modern times, watermarking techniques are being applied to digital documents in order to protect against copyright violations. In these schemes, a content producer who wishes to distribute copies of a digital work marks each copy of the work with a special digital codeword (a fingerprint) and keeps track of which users receive which codewords. If a user illegally shares her copy and the distributor finds this copy, the fingerprint can be extracted from the object and used to reliably identify the perpetrator.

There are many aspects to the watermarking problem, including how to embed the digital fingerprint in the object and how to guarantee that transformations of the object retain the fingerprint. In addition, a good watermarking scheme should protect against *collusion*, that is, groups of users working in unison to thwart the scheme. By comparing their copies of an object and determining how they differ, a group of users might be able to create a new version by combining different pieces of their copies, thereby eliminating the distributor's ability to trace the object back to any of them. Hence, it is important to understand how to construct fingerprinting schemes that are secure against coalitions of attackers.

The literature on this type of fingerprinting is extensive. Blakely, Meadows and Purdy [1] presented early models of fingerprinting in 1985. Kilian et al. [7] propose an orthogonal approach that focuses on reliably embedding bits into documents.

Boneh and Shaw [3] address the explicit goal of constructing a fingerprinting scheme (in effect, a code) for which the distributor can always discover at least one member of any coalition of at most c users who produce an illegal copy. They show that there is no unconditional way to achieve this goal, i.e., a group of collaborators can use a "majority" approach to create codewords which defy unconditionally correct identification. Boneh and Shaw go on to define *collusion-secure codes*, which tolerate an ϵ probability of error by the tracing algorithm against coalitions of size c, and then present constructions of such codes.

The Boneh-Shaw code is used as a building block for many sophisticated digital watermarking schemes, including asymmetric fingerprinting [9] and anonymous fingerprinting [10, 4]. Hence, improving the Boneh-

*Emails: {cpeikert,abhi,asmith}@theory.lcs.mit.edu

Shaw construction may also contribute to many interesting applications.

Because a fingerprint must be embedded into a digital work, its length is an important parameter (robust embedding procedures typically require a large amount of source data for each embedded bit). For $c+1$ total users, the Boneh-Shaw code uses $\Theta(c^3 \log(1/\epsilon))$ bits to attain security against coalitions of size c. For n users where n may be exponential in c, a concatenation scheme [5] yields a code of length $\Theta(c^4 \log n \log(1/\epsilon))$. These lengths become impractical on the Internet, where peer-to-peer services could make c as large as 100 or more. Lindkvist [8] describes constant factor improvements to the length of the Boneh-Shaw code for small values of c. To our knowledge, the Boneh-Shaw code remains the most efficient collusion-secure primitive and there have been no significant improvements in its length. Boneh and Shaw's lower bound proof [3] of $\frac{1}{2}(c-3)\log\frac{1}{c\epsilon}$ on the length of any collusion-secure code describes a strategy that would break any code of shorter length. Thus, there is a substantial gap between the best lower bound and the known constructions.

This paper seeks to address this gap in three ways. First, we show that the Boneh-Shaw construction is as short as it can be by presenting a coalition strategy that breaks the Boneh-Shaw code if the length is any shorter than prescribed.

In Section 5, we then show that any secure code has length $\Omega(c^2 \log(1/c\epsilon))$ when $\log(1/\epsilon) \geq Kk \log c$, for some constant K. Here k is the number of different "column types" appearing in the code, and is some measure of the combinatorial complexity of the code. This implies that no secure code can have length $o(c^2 \log(1/c\epsilon))$, improving the current bound by a factor of c.

Finally, in Section 6, we define a general paradigm for constructing codes: the center chooses a subset V of the hypercube $\{0,1\}^{c+1}$, where c is the coalition size. It constructs the code by repeating each of these column types d times, and randomly permuting them. The success probability of any adversarial strategy will vanish to zero as $d \to \infty$. However, we show that no secure scheme following the paradigm has length $o(\frac{c^3}{\log c}\log(1/c\epsilon))$, by showing a lower bound for the range $\ln(1/\epsilon) \geq K \cdot E(V) \log c/c$, where $E(V)$ is the number of edges within the subgraph induced by V, and K is a constant. This shows that the combinatorial structure of the code is important to the code's performance, since our proof requires a non-trivial bound on the average degree of induced subgraphs in the hypercube.

It also suggests that any attempts at improved constructions should be directed toward techniques that lie outside our framework. We leave open the question of constructing efficient codes for smaller values of $\ln(\frac{1}{\epsilon})$.

2 Notation and Definitions

A *word (of length ℓ)* is an element of $\{0,1\}^\ell$. The value at coordinate i of a word w is denoted by $(w)_i$. For a word w and a set $B = \{i_1, \ldots, i_t\}$, the *restriction of w to B*, denoted $w|_B$, is the word $((w)_{i_1}, \ldots, (w)_{i_t})$. The *weight* of a word w, denoted by $\mathrm{wt}(w)$, is the sum of its entries.

An *(ℓ, n)-code* is a set of n codewords, each of length ℓ. An *(ℓ, n)-coding scheme* is a randomized algorithm for generating an (ℓ, n)-code and some auxiliary output (which will be given to a tracing algorithm).

Suppose we are given a set of words $C = \{u_1, \ldots, u_s\}$. A *column type* is an element of $\{0,1\}^s$. The *column at position i* of C is the word $v_i^C = ((u_1)_i, (u_2)_i, \ldots, (u_s)_i)$. Given a column type $u \in \{0,1\}^s$, define $B_u^C = \{i : v_i^C = u\}$, that is, the set of positions i whose columns are of type u. We omit the superscripts when they are clear from context.

We use $\lg x$ to denote the base two logarithm of x. $H(x)$ is the binary entropy function $H(x) = -x \lg x - (1-x)\lg(1-x)$.

2.1 Collusion-secure codes

We now formalize the model of a collusive attack against a fingerprinting code. The distributor constructs a code and marks each copy of a digital work with a unique codeword using some watermarking scheme specific to the type of data (e.g., music, video, computer programs). Each bit of the codeword is embedded somewhere in the work, so that a user cannot detect or remove the marks. We would like to formalize this requirement, to abstract the specific marking procedure from the construction of codes. This is done by the so-called *Marking Assumption* [3]. First we require a definition:

DEFINITION 1. *Let $C = \{u_1, \ldots, u_k\}$ be a set of codewords from some code. We say that the position i is* undetectable relative to C *if $(u_1)_i = (u_2)_i = \cdots = (u_k)_i$.*

The Marking Assumption states that if a coalition of users has codewords $C = \{u_1, \ldots, u_k\}$, then the coalition's output word must match the codewords in all undetectable positions. (This model was extended to a more realistic one by Yacobi [12], who showed that similar codes—with a constant improvement in length—work in the new model.) Formally, if w is generated by the coalition, then $(w)_i = (u_1)_i$ for all positions i that are undetectable relative to C. At all detectable positions, the output word may be 0, or 1, or an "erasure," denoted by '?' (however, our attacks will not use erasures). This models the fact that users can find all the differences among their files and can

construct a new version by piecing together different parts in which their copies differ.

Our results apply to *collusion-secure* coding schemes, which were first defined and constructed by Boneh and Shaw [3]. Formally:

DEFINITION 2. (SECURE CODING SCHEME) *An (ℓ, n)-coding scheme $\mathcal{U}$ (with* tracing algorithm $\mathcal{T}$*) is a* secure coding scheme *if, for all sufficiently large c, and for all $\epsilon > 0$, the following condition holds: for all coalition strategies $\mathcal{S}$ and all $C = \{i_1, \ldots, i_c\} \subseteq \{1, \ldots, n\}$,*

$$\Pr[(\{u_1, \ldots, u_n\}, aux) \leftarrow \mathcal{U}(1^c, \epsilon); \\ w \leftarrow \mathcal{S}(\{u_{i_1}, \ldots, u_{i_c}\}) : \mathcal{T}(w, aux) \subseteq C] > 1 - \epsilon$$

where the probability is taken over the random choices of $\mathcal{U}$, $\mathcal{S}$, and $\mathcal{T}$.

(All algorithms are probabilistic and polynomial-time in the lengths of their inputs. Note that $\mathcal{U}$ receives c encoded in unary, and ϵ encoded in binary.)

A note about the binary representation of ϵ: because $\mathcal{U}$ is polynomial-time, the resulting code must have length $poly(\ln(1/\epsilon))$. We view $\ln(1/\epsilon)$ as a "security parameter" for the code, because ϵ represents some negligibly small probability.

3 Tools for Lower Bounds

All of our lower bound arguments have a similar flavor. For some particular type of code, we fix the code length to be less than the desired bound and choose an arbitrary set of $c + 1$ users. We then consider all c-sized coalitions of these users and show how each coalition can, with large enough probability, generate a word that belongs to a certain set of "ideal" words. Because no user is a member of every such coalition, the distributor cannot correctly accuse any user with enough confidence. This intuition is formalized in the following lemma:

LEMMA 3.1. *Suppose that there exists a coalition algorithm $\mathcal{S}$ such that for infinitely many c and some $\epsilon > 0$, and for any invocation of an (ℓ, n)-coding scheme $\mathcal{U}$, the following holds:*

> *Fix an arbitrary set $C = \{u_1, \ldots, u_{c+1}\}$ of $c + 1$ codewords, and let $C_j = C - \{u_j\}$. If there exists some set I of "ideal" words, such that for all $j \in \{1, \ldots, c+1\}$ and all $w \in I$:*
>
> $$\Pr[\mathcal{S}(C_j) = w] \geq (c+1)\epsilon/|I|.$$

Then $\mathcal{U}$ is not a secure coding scheme.

Proof. Consider any tracing algorithm $\mathcal{T}$; we will exhibit a subset of c users from $\{1, \ldots, n\}$ for which $\mathcal{T}$'s error probability is at least ϵ, in violation of Definition 2. Note that this probability must include the random choices of $\mathcal{U}$, whereas in our scenario we have already invoked $\mathcal{U}$. For now, we will show that, conditioned on $\mathcal{U}$'s random choices, $\mathcal{T}$ has error at least ϵ. Assume, for the sake of contradiction, that this is not the case.

Let T_w be the (conditioned) random variable indicating the subset of users accused on word w. Note that for all w, $\sum_j \Pr[j \in T_w] \geq 1$, because the algorithm must always accuse some user, and may accuse several users at once.

By assumption, we have for all C_j:

$$\begin{aligned} \epsilon &> \sum_{w \in I} \Pr[\mathcal{S}(C_j) = w \wedge j \in T_w] \\ &= \sum_{w \in I} \Pr[\mathcal{S}(C_j) = w] \Pr[j \in T_w] \\ &\geq \frac{(c+1)\epsilon}{|I|} \sum_{w \in I} \Pr[j \in T_w] \end{aligned}$$

because the random coins of the coalition and the distributor are independent and by hypothesis. Summing the above expression over all $j \in \{1, \ldots, c+1\}$ and dividing by $(c+1)$, we get:

$$\begin{aligned} \epsilon &> \frac{\epsilon}{|I|} \sum_j \sum_{w \in I} \Pr[j \in T_w] = \frac{\epsilon}{|I|} \sum_{w \in I} \sum_j \Pr[j \in T_w] \\ &\geq \frac{\epsilon}{|I|} \sum_{w \in I} 1 = \epsilon, \end{aligned}$$

which contradicts our assumption. So the (conditioned) error probability of the tracing algorithm is at least ϵ.

We complete the argument as follows: say we fix an arbitrary subset of c users, then invoke $\mathcal{U}$. By hypothesis, for any invocation of $\mathcal{U}$, $\mathcal{T}$'s error is at least ϵ (because the coalition strategy can use the words corresponding to our selected users). Therefore, over the random choices of $\mathcal{U}$, $\mathcal{T}$ still has error at least ϵ, and the code is insecure. □

Our lower bound proofs make use of a particular set of probabilities shown in Figure 1. For $i \in \{0, \ldots, \lfloor c/2 \rfloor\}$, define the probability $r_i = \alpha i^2$, where α is such that $r_{\lfloor c/2 \rfloor} = 1/2$ (we will only need the fact that $\alpha \leq 3/c^2$ for sufficiently large c). Then for $i \in \{\lfloor c/2 \rfloor + 1, \ldots c\}$, define $r_i = (1 - r_{c-i})$. Note that $r_0 = 0$ and $r_c = 1$. In addition, define $\delta_i = (r_i - r_{i-1})$ for $i \in \{1, \ldots, c\}$. The r_i values will be used in our attacks against fingerprinting codes: when a coalition detects a column of weight i, it will flip a coin of bias r_i to determine the output value at that column's position.

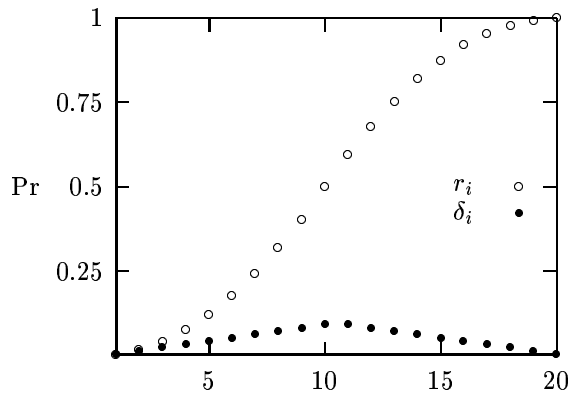

Figure 1: Values of r_i and δ_i for $c = 20$.

Now consider the following experiment: suppose $0 < r < 1$, and we toss a coin having bias r (i.e., the probability of heads is r) m times. Let Δ be such that $0 \le r+\Delta \le 1$ and $t = (r+\Delta)m$ is an integer. Let $B_{r,t,m}$ be the event that exactly t of the tosses are heads. The following lemma proves that this event occurs with good probability.

LEMMA 3.2. *Let $r \in (0,1)$, $r+\Delta \in [0,1]$, and $t \in \{0,\ldots,m\}$, such that $t = (r+\Delta)m$. Then*

$$-\ln \Pr[B_{r,t,m}] \le \frac{m\Delta^2}{\min\{r+\Delta, 1-(r+\Delta)\}} + 2\ln(m+1).$$

Proof. Define $p = r+\Delta = t/m$. Then we have:

$$\begin{aligned}\Pr[B_{r,t,m}] &= \binom{m}{pm} r^{pm}(1-r)^{(1-p)m} \\ &\ge (m+1)^{-2} 2^{mH(p)} r^{pm}(1-r)^{(1-p)m}\end{aligned}$$

by the well-known lower bound for binomial coefficients, $\binom{t}{\beta t} \ge (t+1)^{-2}2^{tH(\beta)}$ [6]. Taking logarithms, we have:

$$\begin{aligned}&\ln \Pr[B_{r,t,m}] + 2\ln(m+1) \\ &\ge mH(p) + pm\ln r + (1-p)m\ln(1-r) \\ &= pm\ln(\frac{r}{p}) + (1-p)m\ln(\frac{1-r}{1-p}) \\ &= pm\ln(1-\frac{\Delta}{p}) + (1-p)m\ln(1+\frac{\Delta}{1-p})\end{aligned}$$

Using the inequality $\ln(1+x) \ge x - x^2/2$, we get:

$$\begin{aligned}&\ln \Pr[B_{r,t,m}] + 2\ln(m+1) \\ &\ge m\left(-\Delta - \frac{\Delta^2}{2p} + \Delta - \frac{\Delta^2}{2(1-p)}\right) \\ &= \frac{-m\Delta^2}{2p(1-p)}.\end{aligned}$$

Suppose without loss of generality that $p \le 1/2 \le 1-p$; then $1/2(1-p) \le 1$ and the claim follows. □

Our next lemma shows that, with some probability that is large enough for our purposes, throwing an r_{i-1}-biased coin has the same outcome as throwing an r_i-biased coin.

LEMMA 3.3. *For any $i \in \{1,\ldots,c\}$, let r_{i-1}, r_i be as described above. Suppose we have a coin with bias either r_{i-1} or r_i, which we flip m times. Then there exists $t_{i,m} \in \{0,\ldots,m\}$ such that with either coin, exactly $t_{i,m}$ heads occur with significant probability:*

$$-\ln \min\left\{\begin{array}{l}\Pr[B_{r_{i-1},t_{i,m},m}],\\ \Pr[B_{r_i,t_{i,m},m}]\end{array}\right\} \le O(m\alpha+1) + 2\ln(m+1)$$

Proof. We first consider the case when $i = 1$ or $i = c$. If $i = 1$, then we let $t_{i,m} = 0$. A coin with bias $r_{i-1} = 0$ always yields $t_{i,m} = 0$ heads, and a coin with bias $r_i = \alpha$ yields zero heads with probability $(1-\alpha)^m$. Then $-\ln(1-\alpha)^m = m\ln\frac{1}{1-\alpha} \le \frac{m\alpha}{1-\alpha} \le 2m\alpha$, as desired. The analysis for $i = c$ is similar, where we let $t_{i,m} = m$. From now on assume that $i \in \{2,\ldots,c-1\}$.

Next we address the case when m is "large," i.e. when $m \ge 1/\delta_i$. In this case, the interval $[mr_{i-1}, mr_i]$ has length at least 1, so it contains some integer $t_{i,m}$. Assume without loss of generality that $r_i \le 1/2$; then by Lemma 3.2, the expression from the claim is at most $m\delta_i^2/r_{i-1} + 2\ln(m+1)$. We have that $\delta_i = r_i - r_{i-1} = \alpha(2i-1)$, so we can bound the fractional term as desired:

$$\frac{m\alpha^2(2i-1)^2}{\alpha(i-1)^2} \le m\alpha\left(\frac{2i-1}{i-1}\right)^2 \le 9m\alpha.$$

Finally we address the case when m is small; that is $m\delta_i < 1$. We use a slightly different technique here: let D_{i-1}, D_i be the distributions corresponding to flipping m coins with bias r_{i-1} and r_i, respectively. Consider the sum over *all* words w in $\{0,1\}^m$ of the minimum of the probability of w arising under either D_{i-1} or D_i. We can bound the probability of each w below by taking the product of the least likelihoods for each of the coordinates of w:

$$\sum_w \min\left\{\Pr_{D_{i-1}}[w], \Pr_{D_i}[w]\right\} \ge \sum_{t=0}^{m}\binom{m}{t} r_{i-1}^t (1-r_i)^{m-t}.$$

This last sum is the binomial expansion of $(r_{i-1} + 1 - r_i)^m = (1-\delta_i)^m$. There must exist some weight t for which the sum over words of weight t yields $(1-\delta_i)^m/(m+1)$. Hence, the negative log of the minimum probability that one gets exactly t heads is $-\ln[(1-\delta_i)^m/(m+1)] \le O(m\delta_i) + \ln(m+1)$. Since $m < 1/\delta_i$, the last bound is $O(1) + \ln(m+1)$. □

4 The Boneh-Shaw Code

As a primitive, Boneh and Shaw [3] construct a secure $(\ell, c+1)$-coding scheme, where $\ell = O(c^3 \log(1/\epsilon))$. This primitive is then concatenated with other codes, resulting in practical constructions whose lengths are logarithmic in the number of users.

The construction begins with a fixed code $\Gamma_0(c,d)$. Let x_i be a column of height $c+1$ in which the first i bits are 1, and the rest are 0. Then $\Gamma_0(c,d)$ consists of the columns $x_1, \ldots, x_c$ each duplicated d times. For example, when $c = 4$ the code $\Gamma_0(4,d)$ for 5 users is shown in Figure 2. The security comes from a secret permutation $\pi \in S_{cd}$ that is applied to each word, to create the code used for marking. Boneh and Shaw prove that if $d \geq 2c^2 \log(2c/\epsilon)$, then the code is secure.

$$\begin{array}{lcccccc} & & \overbrace{}^{d} & \overbrace{}^{d} & \overbrace{}^{d} & \overbrace{}^{d} \\ u_1 & = & 1\cdots1 & 1\cdots1 & 1\cdots1 & 1\cdots1 \\ u_2 & = & 0\cdots0 & 1\cdots1 & 1\cdots1 & 1\cdots1 \\ u_3 & = & 0\cdots0 & 0\cdots0 & 1\cdots1 & 1\cdots1 \\ u_4 & = & 0\cdots0 & 0\cdots0 & 0\cdots0 & 1\cdots1 \\ u_5 & = & 0\cdots0 & 0\cdots0 & 0\cdots0 & 0\cdots0 \end{array}$$

Figure 2: The Boneh-Shaw code $\Gamma_0(4,d)$, before applying a permutation.

Recall that the set B_{x_k} is the set of positions in the code whose columns are x_k. The security of the Boneh-Shaw construction relies on the fact that if user u_j is innocent, the coalition cannot distinguish $B_{x_{j-1}}$ from B_{x_j}. If the coalition outputs a word w and d is large enough, with high probability $\mathrm{wt}(w|_{B_{x_{j-1}}}) \approx \mathrm{wt}(w|_{B_{x_j}})$. By contrapositive, if these weights differ significantly, this indicates that user j is probably guilty. Also note that user 1 is certainly guilty if $\mathrm{wt}(w|_{B_{x_1}}) \neq 0$, and user c is certainly guilty if $\mathrm{wt}(w|_{B_{x_c}}) \neq d$. Because of these two endpoint constraints, the weights $\mathrm{wt}(w|B_{x_i})$ must increase from 0 to d as i goes from 1 to c, so there must be a large gap between two adjacent weights. This guarantees that the tracing algorithm can always find a user who is guilty with high probability.

4.1 A Tight Lower Bound Our lower bound states that the Boneh-Shaw construction is tight, i.e., that d must be (asymptotically) as large as specified in their construction, or else the code is insecure. The proof is relatively simple, but it provides an instructive warm-up for our more general results. In attacking a Boneh-Shaw code, we cannot escape the endpoint constraints that force the block weights to increase from 0 to d. However, if the code is relatively short, we *are* able to create words whose block weights increase "smoothly," so that the gaps between adjacent weights are not too large. The shape of these ideal block weights are in fact given by the $t_{i,d}$ values described in Lemma 3.3. When the weights increase in this way, no tracing algorithm is able to determine the guilt of any single user with enough confidence.

THEOREM 4.1. *Let $\mathcal{B}$ be the Boneh-Shaw coding scheme, that is, the $(cd, c+1)$-coding scheme which starts with $\Gamma_0(c+1,d)$ and permutes the columns randomly. If $\ell = o(c^3 \log(\frac{1}{c\epsilon}))$, then $\mathcal{B}$ is not a secure coding scheme.*

Proof. First we define the set of ideal words for the code. Recall that the Boneh-Shaw code is a random permutation of d copies of the columns types $x_1, \ldots, x_c$. The ideal words are those that have weight $t_{i,d}$ when restricted to columns of weight i, for all i (where $t_{i,d}$ is the ideal weight from Lemma 3.3). Formally, the set of ideal words is:

$$I = \{w \,:\, \forall\, i \in \{1, \ldots, c\},\ \mathrm{wt}(w|_{B_{x_i}}) = t_{i,d}\}.$$

Let the code be $C = \{u_1, \ldots, u_{c+1}\}$, and define the coalition $C_j = C - \{u_j\}$. Now we describe a coalition algorithm for creating an output word, w, and show that w is an ideal word with large enough probability. There are two special cases: when the algorithm is given C_1 (i.e., the missing word's entries are all 1s), and when it is given C_{c+1} (i.e., the missing word's entries are all 0s). In the former case, the algorithm cannot detect positions B_{x_1}, so it must write 0s in those positions (and because $t_{1,d} = 0$, $\mathrm{wt}(w|_{B_{x_1}}) = t_{1,d}$ as desired). In the latter case, the algorithm cannot detect positions B_{x_c}, so it must write 1s in those positions ($t_{c,d} = d$, so $\mathrm{wt}(w|_{B_{x_c}}) = t_{c,d}$). In either case, the algorithm detects d columns each having weights $1, 2, \ldots, c-1$, so it cannot tell whether those sets of columns are, respectively, $\{B_{x_2}, \ldots, B_{x_c}\}$, or $\{B_{x_1}, \ldots, B_{x_{c-1}}\}$. Therefore the algorithm simply guesses which situation it is in. More specifically, it chooses $b \in \{0,1\}$ at random and then randomly places exactly $t_{i+b,d}$ 1s in the weight i columns, for each i. In these two cases, w is ideal with probability 1/2.

In the general case, the coalition algorithm is given C_j for some $j \neq 1, c+1$. The coalition can therefore discover j, by detecting exactly d columns each of weight 1 through $j-2$, exactly $2d$ columns of weight $j-1$, and exactly d columns each of weights j through $c-1$. Those columns having weights 1 through $j-2$ comprise positions B_{x_1} to $B_{x_{j-2}}$, respectively, so the algorithm randomly puts the proper number of ones in those positions. Similarly, the columns having weights j through c comprise positions $B_{x_{j+1}}$ to B_{x_c}, respectively, so the algorithm randomly places the proper number of

ones at those positions. For each of the $2d$ positions in $B_{x_{j-1}} \cup B_{x_j}$, the algorithm flips a coin with bias r_{j-1} and writes a one for heads, and zero otherwise.

First note that when the output word w is ideal (i.e., $w \in I$), it is uniformly distributed over I since the coalition's coin flips are independent. It now suffices to show that $\Pr[w \in I] \geq (c+1)\epsilon$. Lemma 3.1 then implies that the code is not secure.

Suppose that $\ell = o(c^3 \ln(1/c\epsilon))$; then $d = o(c^2 \ln(1/c\epsilon))$. We may assume that $d+1 \leq ((c+1)\epsilon)^{-1/8}$, because d must be polynomial in $\ln 1/\epsilon$. Recall that $\alpha \leq 3/c^2$. By the description of the coalition algorithm, $w \in I$ if events $B_{r_{j-1},t_{j-1,d},d}$ and $B_{r_{j-1},t_{j,d},d}$ occur. By Lemma 3.3,

$$\begin{aligned} -\ln \Pr[w \in I] &= O(d\alpha + 1) + 4\ln(d+1) \\ &\leq o\left(\ln \frac{1}{(c+1)\epsilon}\right) + \frac{1}{2}\ln \frac{1}{(c+1)\epsilon} \\ &\leq \ln \frac{1}{(c+1)\epsilon}, \end{aligned}$$

for sufficiently large c and small ϵ, so $\Pr[w \in I] \geq (c+1)\epsilon$ and we are done. □

5 A General Lower Bound

In this section, we extend the lower bound from the previous section to apply to any secure coding scheme. With Boneh-Shaw codes, a coalition only tosses coins for $2d$ indistinguishable columns since it can detect and categorize all the other columns exactly. In the general case, the coalition may not be able to categorize any of the columns exactly, so it instead tosses a coin of appropriate bias for every column. As a result, the lower bound loses a factor of c.

Furthermore, the success of this strategy depends on the number of different column types which appear in a particular coding scheme—the fewer there are, the better the attack's performance. For a coding scheme $\mathcal{U}$, let k denote the expected number of different column types (taken over the distributor's random choices).

THEOREM 5.1. *If $\mathcal{U}$ is an (ℓ, n)-coding scheme, then $\ell = \Omega(c^2 \log(\frac{1}{c\epsilon}))$ when $\ln(\frac{1}{\epsilon}) \geq K \cdot k \log c$, where k is the expected number of distinct column types and K is some large enough constant.*

In particular, this means that no secure code has $\ell = o(c^2 \log(\frac{1}{c\epsilon}))$.

Proof. Coalition Strategy. We show that a simple coalition strategy works against all codes: For each position i, let y be the number of ones seen by the coalition in that position. The coalition flips a coin having bias r_y and writes 1 for heads, and 0 for tails (if the coalition cannot detect a column because it is all 0s or all 1s, the coalition behaves as described, because $r_0 = 0$ and $r_c = 1$).

The code used by the center depends on its random choices, and hence so does the set of ideal words we use for the lower bound proof. We first analyze the case in which the center has no randomness, that is the code itself is fixed. For a particular setting ρ of the center's randomness, let V_ρ denote the corresponding set of columns. Let $k_\rho = |V_\rho|$ denote the number of different column types which appear, and let ϵ_ρ be the conditional probability of the coalition being successful given the coins ρ. Note that the coalition's global success probability is the expected value of ϵ_ρ taken over the choices for ρ: $\epsilon = \mathbb{E}_\rho[\epsilon_\rho]$. Similarly, we use $k = \mathbb{E}_\rho[k_\rho]$ to denote the expected number of column types.

Ideal Target Words. Pick an arbitrary $(c+1)$-subset $C = \{u_1, \ldots, u_{c+1}\}$ of $\mathcal{U}$, and let C_j be the size c coalition $C_j = C - \{j\}$. When omitted, all references to columns will be with respect to C. The ideal words are similar to those in Theorem 4.1: they have weight exactly $t_{y,|B_x|}$ when restricted to B_x, for each column type x of weight y (recall that B_x is the set of positions in which the values of words in C form the vector x). Formally:

$$I = \{w \ : \ \forall\, x \in \{0,1\}^{c+1},\ \mathrm{wt}(w|_{B_x}) = t_{\mathrm{wt}(x),|B_x|}\}$$

LEMMA 5.1. *Let $V_\rho, k_\rho, \epsilon_\rho$ be as in the previous discussion. The coin-flipping coalition strategy implies*

$$-\ln(\epsilon_\rho(c+1)) \leq O(\alpha\ell) + 3k_\rho \ln(\ell/k_\rho)$$

We prove this lemma below. For now, note that this bound depends only on the number of column types, and has no relation to the actual code. This bound still holds when we average over the choice of ρ. We get:

$$\epsilon(c+1) \geq \mathbb{E}_\rho\left[e^{-O(\alpha\ell) - 3k_\rho \ln(\ell/k_\rho)}\right] = e^{-O(\alpha\ell)}\mathbb{E}_\rho[g(k_\rho)],$$

where $g(k) = e^{-3k\ln(\ell/k)}$. One can verify by computing the second derivative that $g(k)$ is convex-$\cup$ in k for all settings of ℓ, and hence we can apply Jensen's inequality to obtain:

$$\epsilon(c+1) \geq e^{-O(\alpha\ell)} g(k).$$

This means that one of the two terms in the product must be smaller than $\sqrt{\epsilon(c+1)}$ in order for the code to be secure. Now the first possibility, $e^{-O(\alpha\ell)} \leq \sqrt{\epsilon(c+1)}$, implies $\ell \geq \Omega(\frac{1}{\alpha}\ln\frac{1}{\epsilon(c+1)}) = \Omega(c^2 \ln\frac{1}{\epsilon(c+1)})$ (this is the bound we claim in the theorem statement).

The second possibility, namely $g(k) \leq \sqrt{\epsilon(c+1)}$, holds, in particular, when $\ell \geq k\exp(\frac{1}{6k}\ln\frac{1}{\epsilon(c+1)})$. Thus:

$$\ell \in \Omega\left(\min\left\{c^2 \log\frac{1}{c\epsilon}\ ,\ k\exp(\frac{1}{6k}\log\frac{1}{\epsilon(c+1)})\right\}\right).$$

Finally, this minimum is dominated by its left-hand argument when $\ln(\frac{1}{\epsilon}) \geq K \cdot k \log c$ for some large enough constant K. This completes the proof of Theorem 5.1. All that is left is to prove Lemma 5.1. □

Proof. [Lemma 5.1] We would like to apply Lemma 3.1. First, note that within the set I, the coalition's output is uniformly distributed, since the coalition's coin flips are independent.

Now fix a particular coalition C_j. For each possible column type $x \in \{0,1\}^{c+1}$ where $y = \mathrm{wt}(x)$, we can apply Lemma 3.3, since we know that C_j flips coins with bias either r_y or r_{y-1}. The coalition generates an ideal word if it generates a word of weight $t_{\mathrm{wt}(x),|B_x|}$ on each block B_x.

Hence the (negative natural log of the) probability that coalition C_j creates an ideal word is therefore

$$\begin{aligned} & -\ln \Pr[w \in I] \\ & \leq \sum_{x \in \{0,1\}^{c+1}} \Big(O(\alpha |B_x| + 1) + 2\ln(|B_x|) \Big) \\ & \leq O(\alpha \ell) + 3 \sum_x \ln |B_x| \end{aligned}$$

The summation is maximized when all the $|B_x|$ are equal, since they must sum to ℓ. Because there are k_ρ unique column types, the summation is at most $k_\rho(\ln(\frac{\ell}{k_\rho}))$. Moreover, by Lemma 3.1, the conditional error probability ϵ_ρ satisfies $\ln(c+1)\epsilon_\rho \geq \ln \Pr[w \in I]$. Hence we get $-\ln(c+1)\epsilon_\rho \leq O(\alpha\ell) + 3k_\rho(\ln(\ell/k_\rho))$. □

6 Bounds for the Multiplicity Paradigm

In this section, we prove a lower bound for any coding scheme that follows a certain general approach, which we call the "multiplicity paradigm." The Boneh-Shaw construction, in particular, fits this paradigm. The essential properties are as follows: each column is duplicated d times (its "multiplicity"), where d depends on the desired security ϵ, while the choice of the columns themselves is independent of ϵ. Moreover, the number of indistinguishable columns is the same for every coalition and the security of the scheme depends entirely upon the secrecy of the permutation π applied to the duplicated columns (not on the secrecy of the columns themselves).

For a set of columns V, define $E^{(j)}(V) = \sum_{x \in V} \sum_{y \in V} I^{(j)}_{x,y}$, where $I^{(j)}_{x,y}$ is the indicator variable that is 1 if x and y differ only in coordinate j, and 0 otherwise. Also, define $E(V) = \sum_j E^{(j)}(V)$. If we view V as a choice of vertices defining a subgraph of the hypercube of dimension $c+1$, then $E(V)$ is the number of edges in that subgraph. We are now ready to make a formal definition of the paradigm, and to prove a lower bound.

DEFINITION 3. (MULTIPLICITY PARADIGM) *An $(\ell, c+1)$-coding scheme $\mathcal{U}$ follows the* multiplicity paradigm *with* columns $\{V_c\}$ *(we call $\mathcal{U}$ a* multiplicity code*) if the following hold:*

1. *$\{V_c\}$ is a (polynomial-sized) publicly-known ensemble, indexed over c, where V_c is a code of length $c+1$ representing the columns.*
2. *For all j, $E^{(j)}(V_c) = O(1/c)E(V_c)$.*
3. *$\mathcal{U}(1^c, \epsilon)$ constructs the output code by randomly permuting d copies of each column of V_c, and letting the rows be the codewords.*

THEOREM 6.1. *Let $\mathcal{U}$ be an $(\ell, c+1)$-coding scheme that follows the multiplicity paradigm with columns $\{V_c\}$. Then $\ell = \Omega(\frac{c^3}{\log |V_c|} \log \frac{1}{c\epsilon})$ when $\ln(\frac{1}{\epsilon}) \geq K \max\{(E(V_c)\log c)/c,\ c \log c\}$ for some constant K.*

In particular, no secure multiplicity code has length $\ell = o(\frac{c^3}{\log c} \log \frac{1}{c\epsilon})$.

Proof. Our proof is a combination of techniques from Theorems 4.1 and 5.1. The ideal words again have weight $t_{\mathrm{wt}(x),|B_x|}$ when restricted to B_x, for all $x \in V_c$.

The coalition strategy relies on the fact that when the column set is public, there are relatively few columns for which the missing user's value is unknown. First, the coalition guesses the proper ordering of its codewords and determines where the missing codeword fits in the ordering. This can be done with probability $\frac{1}{(c+1)!}$. As in Theorem 4.1, when the coalition detects exactly d copies of a certain column type, it then checks V_c for the matching (and complete) column type x, and places $t_{\mathrm{wt}(x),|B_x|}$ 1s on those columns (if there is no match, then the coalition has guessed the ordering of its users incorrectly, and simply gives up by outputting one of its users' codewords). If the coalition detects exactly $2d$ columns of a certain type x, then it flips a coin having bias $r_{\mathrm{wt}(x)}$ for each of the $2d$ positions. Coalition C_j sees $E^{(j)}(V_c)$ such blocks of $2d$ indistinguishable columns. As in Theorem 5.1, for coalition C_j:

$$\begin{aligned} & -\ln \Pr[w \in I] \\ & = O\Big(c \ln c + d\alpha E^{(j)}(V_c) + E^{(j)}(V_c)\ln(d+1)\Big) \\ & = O\Big(c \ln c + E^{(j)}(V_c)(d/c^2 + \ln d)\Big) \end{aligned}$$

Because $\ell = d|V_c|$, and by Definition 3, we can simplify this to

$$O\Big(c \ln c + E(V_c)[\ell/(c^3|V_c|) + \frac{1}{c}\ln(\ell/|V_c| + 1)]\Big).$$

As in Theorem 5.1, we can set this equal to $\ln((c+1)\epsilon)$. We obtain three inequalities, since one of the terms in

this sum must exceed $\frac{1}{3}\ln((c+1)\epsilon)$ for the code to be secure. If we assume that $\ln(\frac{1}{\epsilon}) \geq 3c\log c$, we obtain a lower bound for ℓ from the remaining two inequalities:

$$\ell = \Omega\left(\min\left\{\begin{array}{l}\frac{|V_c|}{E(V_c)}c^3\log\frac{1}{c\epsilon},\\ |V_c|\exp\left(O\left(\frac{c}{E(V_c)}\log\frac{1}{c\epsilon}\right)\right)\end{array}\right\}\right)$$

The condition $\ln(\frac{1}{\epsilon}) \geq K(E(V_c)\log c)/c$ makes the top term of the minimum dominate. (To see this, set $f = \frac{c\log(1/c\epsilon)}{E(V_c)}$. Then the top argument dominates when $c^2 \leq \frac{1}{f}\exp(O(f))$, which is true when $f \geq K'\log c)$ for some suitably large K'.) Ultimately, we get that the code is insecure if $\ell = o\left(\frac{|V_c|}{E(V_c)}c^3\ln\frac{1}{c\epsilon}\right)$.

To make this bound concrete, we need to upper-bound the average degree of a subgraph of the hypercube of dimension $c+1$. It is known that the total number of edges having both endpoints in V_c is at most $\frac{1}{2}|V_c|\lceil\lg|V_c|\rceil$ ([2], Section 16). Since $|V_c|$ is independent of ϵ, and hence polynomial in c by Definitions 2 and 3, the average degree is $O(\log c)$, yielding the desired result. □

7 Conclusions and Open Problems

We have tightened the gap between the best known construction and lower bound for general secure codes, but a factor of c still remains. Additionally, we have proven bounds only for extremely low error rates. Whether these bounds extend to practical values of ϵ is unknown. Resolving these gaps is an important question.

We have also defined a wide class of code constructions, and have given better lower bounds on their length. A $\log c$ gap remains between the Boneh-Shaw construction and our lower bound. We conjecture that the following construction resolves the issue. Choose column types in the following way: starting with $\vec{0}$, let the first $\log c$ coordinates take on all $O(c)$ possible values. Then, with the first $\log c$ coordinates set to 1, let the next $\log c$ coordinates take on all values. Repeat $c/\log c$ times, for a total of $O(c^2)$ columns types and average degree $\Theta(\log c)$. When this subgraph is used in the multiplicity paradigm, the resulting code is of length $O(\frac{c^3}{\log c}\log(1/\epsilon))$. We note that our coalition attack from Theorem 6.1 does *not* work against this construction; however, it still remains to devise a tracing algorithm and bound its error.

Other avenues for constructing more efficient codes should focus on breaking the multiplicity paradigm in some way, for example, by directly constructing codes for more than $c+1$ users, randomly (and secretly) choosing the columns, or varying the multiplicities of the columns.

8 Acknowledgements

We would like to thank Eric Lehman for introducing the problem and Venkat Guruswami for pointing out the upper bound on the average degree of a subgraph of the hypercube. Dan Boneh provided very useful feedback on this work, and also helped us improve the presentation of the results. Finally, we would like to thank Madhu Sudan and Ron Rivest for support and helpful comments. A. Smith was supported by U.S. ARO Grant DAAD19-00-1-0177.

References

[1] G.R. Blakely, C. Meadows, and G.B. Purdy. "Fingerprinting long forgiving messages," In Hugh C. Williams, editor, Advances in Cryptology – CRYPTO'85, LNCS 218, Springer-Verlag, Berlin, 1985, p. 180.

[2] B. Bollobás. *Combinatorics.* Cambridge University Press, 1986.

[3] D. Boneh and J. Shaw. "Collusion-secure fingerprinting for digital data," IEEE Transactions on Information Theory, vol IT-44, Sep. 1998, pp. 1897–1905.

[4] J. Camenisch. "Efficient Anonymous Fingerprinting with Group Signatures." In T. Okamoto, editor, ASIACRYPT 2000, LNCS 1976, pp. 415–428.

[5] B. Chor, A. Fiat, and M. Naor. "Tracing traitors." In Yvo G. Desmedt, editor, Advances in Cryptology – CRYPTO'94, LNCS 839, Springer-Verlag, Berlin, 1994, pp. 257–270.

[6] I. Csiszár and J. Körner, *Information Theory. Coding Theorems for Discrete Memoryless Systems*, Academic Press, New York, 1982.

[7] Joe Kilian, F. Thomson Leighton, Lesley R. Matheson, Talal G. Shamoon, Robert E. Tarjan, and Francis Zane. "Resistance of digital fingerprints to collusional attacks." Proceedings of 1998 IEEE International Symposium on Information Theory. Cambridge, MA, 16-21 August 1998, p. 271.

[8] Tina Lindkvist. "Fingerprinting of digital documents," Dissertation No 706, Linköping University, September 2001.

[9] B. Pfitzmann and M. Waidner. "Asymmetric fingerprinting for larger collusions." In 4th ACM Conference on Computer and Communication Security, 1997.

[10] B. Pfitzmann and M. Waidner. "Anonymous fingerprinting." In Walter Fumy, editor, Advances in Cryptology – EUROCRYPT'97, LNCS 1233, Springer-Verlag, Berlin, 1997, pp. 88-102.

[11] A. Silverberg and J. Staddon and J. L. Walker. "Efficient traitor tracing algorithms using list decoding," In Colin Boyd, editor, Advances in Cryptology – ASIACRYPT 2001, LNCS 2248, Springer-Verlag, Berlin, 2001, pp. 175–192.

[12] Y. Yacobi. "Improved Boneh-Shaw Content Fingerprinting," Proceedings of CT-RSA 2001, San Francisco, CA, pp. 378–391.