Correcting Errors Without Leaking Partial Information

Yevgeniy Dodis*
New York University
dodis@cs.nyu.edu

Adam Smith†
Weizmann Institute of Science
adam.smith@weizmann.ac.il

ABSTRACT

This paper explores what kinds of information two parties must communicate in order to correct errors which occur in a shared secret string W. Any bits they communicate must leak a significant amount of information about W — that is, from the adversary's point of view, the entropy of W will drop significantly. Nevertheless, we construct schemes with which Alice and Bob can prevent an adversary from learning any *useful* information about W. Specifically, if the entropy of W is sufficiently high, then there is no function $f(W)$ which the adversary can learn from the error-correction information with significant probability. This leads to several new results: (a) the design of noise-tolerant "perfectly one-way" hash functions in the sense of Canetti *et al.* [7], which in turn leads to obfuscation of proximity queries for high entropy secrets W; (b) private fuzzy extractors [11], which allow one to extract uniformly random bits from noisy and nonuniform data W, while also insuring that no sensitive information about W is leaked; and (c) noise tolerance and stateless key re-use in the Bounded Storage Model, resolving the main open problem of Ding [10].

The heart of our constructions is the design of strong randomness extractors with the property that the source W can be recovered from the extracted randomness and any string W' which is close to W.

Categories and Subject Descriptors

E.4 [**Data**]: Coding and Information Theory; G.2.1 [**Discrete Mathematics**]: Combinatorics; H.1.1 [**Systems and Information Theory**]: Information Theory

*Supported by NSF CAREER award 0133806 and Trusted Computing grant 0311095.

†This work done while the author was a student at the MIT CS and AI Lab, supported by the US DoD MURI program administered by the Army Research Office under grant DAAD19-00-1-0177 and by a Microsoft graduate fellowship.

STOC'05, May 22-24, 2005, Baltimore, Maryland, USA.

General Terms

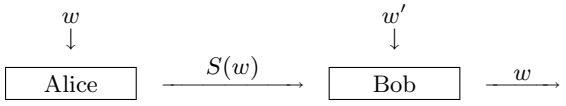
Theory, Security, Algorithms

Keywords

Randomness Extractors, Error-Correcting Codes, Information Reconciliation, Cryptography, Bounded Storage Model, Entropic Security, Code Obfuscation

1. INTRODUCTION

This paper investigates what kind of information must be leaked to an eavesdropper when two cooperating parties communicate in order to correct errors in a shared secret string.



Suppose that Alice and Bob share an n-bit secret string. Alice's copy w of the shared string is slightly different from Bob's copy w'. Alice would like to send a short message $S(w)$ to Bob which allows him to correct the errors in w' (and thus recover w) whenever w and w' differ in at most τ bits. The randomized map $S()$ that Alice applies to w to get the message she sends to Bob is called a *non-interactive information reconciliation scheme*, or simply a *sketch*, correcting τ errors. A typical example of a sketch is

$$S(w) = \mathsf{syn}_C(w),$$

where syn_C is the syndrome of a linear error-correcting code C with block length n (see below for definitions) [2]. If C has dimension k, then $\mathsf{syn}_C(w)$ is only $n-k$ bits long. If the minimum distance of C is at least $2\tau+1$, then $\mathsf{syn}_C(w)$ allows Bob to correct any τ errors in w'. Moreover, the process is efficient if the code can correct τ errors in polynomial time.

Enter Eve, who is tapping the line and trying to learn as much as possible. From her point of view, Alice and Bob hold a pair of random variables W, W'. Suppose that Alice and Bob do not share any secrets except this pair (this rules out trivial solutions, such as Alice sending the encryption of W with Bob's public key). What kind of guarantees can Alice and Bob obtain on what Eve learns from seeing $S(W)$? Standard notions of security do not fit here. The statement "$S(W)$ leaks no information about W" is normally formalized by requiring that W and $S(W)$ be almost statistically independent or, equivalently, that the Shannon mutual information $\mathbf{I}(W; S(W))$ be very small. Such a strong

requirement is impossible to achieve in our setting: a coding argument shows that the mutual information must be large (e.g., larger than τ) in general [4]. Even the analogue requirement for computationally bounded adversaries, *semantic security* [13], is impossible here: if Eve knows that W is one of two strings w_1, w_2 which differ in only a few bits, then she can use whatever algorithm Bob would have run to compute W from $S(W)$ and w_1.

The difficulty, then, is that the standard definitions of security require secrecy even when Eve knows a lot about W. We show that when this requirement is relaxed (that is, when Eve is sufficiently uncertain about W), a strong secrecy guarantee can be provided.

A more suitable definition for our setting is *entropic security* [7, 24, 12]. If W, Y are (correlated) random variables, Y *hides all functions of* W if for every function f, it is nearly as hard to predict $f(W)$ given Y as it is without Y, regardless of the adversary's computing power. A randomized map $S()$ is called *entropically secure* if $S()$ hides all functions of W whenever the min-entropy[1] of W is above a certain threshold. This definition of security has already produced surprising results in two contexts. Canetti, Micciancio and Reingold [6, 7] constructed hash functions whose outputs leak no partial information about the input. Russell and Wang [24] and Dodis and Smith [12] gave entropically secure symmetric encryption schemes with keys much shorter than the length of the input, thus circumventing Shannon's famous lower bound on key length.

This paper introduces a third, very different application of entropic security: we construct secure sketches that are (a) efficiently decodable (that is, Bob's recovery algorithm is polynomial-time) and (b) entropically secure. In particular, for any entropy bound t which is linear in n, we obtain sketches which can efficiently decode a constant fraction of errors and have leakage exponentially small in n. The core of our construction is a family of strong *randomness extractors* with an additional property: given the output of the extractor and a string which is close to the source, one can efficiently recover the source exactly. We construct these extractors based on small random families of algebraic-geometric codes, and then apply our constructions to private storage of keys derived from biometric measurements, obfuscation of proximity queries, and key re-use in the bounded storage model (see Section 1.1).

The Relation to Entropy Loss. The task of correcting errors in a joint string is usually called *information reconciliation* [2, 4, 5, 17, 10], *fuzzy cryptography* ([15], see [26] for a survey), or *document exchange* (in communication complexity, e.g. [9, 8]). In contrast to this paper, previous work focused only on maximizing the length of a cryptographic key which can be derived from W once the errors in W' have been corrected. Because of that, they are only interested in bounding the drop in the entropy of W from Eve's point of view when she sees the communication between Alice and Bob.

The security guarantee we provide is strictly stronger than in previous work. Indeed, min-entropy is an upper bound on all the measures of entropy used in the literature, and entropic security implies a lower bound on the min-entropy of W given the sketch $S(W)$. The converse implication is not true: simply bounding the min-entropy of W given the sketch does not prevent Eve from learning some particular function of W with probability 1 (for example, the syndrome construction above always reveals a particular, fixed set of linear combinations of the bits of W). This can be a problem for several reasons. First, W itself may be sensitive (say, if it is a biometric used for authentication [15, 16, 11]), in which case $S(W)$ might reveal sensitive information, such as a person's age. Second, when we use the error-correction protocol as a piece of a larger framework, entropy loss may not be a sufficient guarantee of secrecy; we will see an example of this in key agreement protocols which are secure against memory-bounded adversaries [10].

For completeness, we state the min-entropy loss of our constructions explicitly, since it is typically much lower than the bound implied by entropic security.

Notation and Definitions. We denote the output of a randomized algorithm on input x and random coins r by $Y(x;r)$. We use the shorthand $Y(x)$ for (random) output when the string r is chosen uniformly at random. The *statistical difference* between two probability distributions A and B on the same space is $\mathbf{SD}(A,B) \stackrel{\text{def}}{=} \frac{1}{2}\sum_v \left|\Pr[A=v] - \Pr[B=v]\right|$ (that is, half the L_1 distance between the probability mass functions). The main measure of entropy we use is *min-entropy*, which measures the difficulty of guessing a random variable A a-priori: the best predictor succeeds with probability $p^* = \max_a \Pr[A=a]$, and the min-entropy is $\mathbf{H}_\infty(A) = -\log(p^*)$ (all logarithms are base 2 by default). A is called a t-source if $\mathbf{H}_\infty(A) \geq t$. The conditional min-entropy of A given B is $\bar{\mathbf{H}}_\infty(A \mid B) \stackrel{\text{def}}{=} -\log(\mathbb{E}_{b \leftarrow B}\left[2^{-\mathbf{H}_\infty(A|B=b)}\right])$. (This definition is not standard but very convenient.) Finally, $h_2()$ denotes the binary entropy function.

We now turn to defining secure sketches and entropic security. Following [11], we incorporate entropy loss into the definition of a secure sketch; we state the definition of entropic security separately.

Definition 1 ([11]). *A (t, t', τ)-*secure sketch *is a pair of (possibly) randomized maps $S : \{0,1\}^n \to \{0,1\}^*$ and $\mathsf{Rec} : \{0,1\}^* \to \{0,1\}^n$ such that:*

- *For all pairs of strings w, w' of distance at most τ, we have $\mathsf{Rec}(w', S(w)) = w$ with probability 1.*
- *For all t-sources W, we have $\bar{\mathbf{H}}_\infty(W \mid S(W)) \geq t'$.*

The entropy loss *of a sketch is the difference $t - t'$.*
The sketch is efficient *if S and Rec run in time poly(n).*

Definition 2 ([7, 24, 12]). *The probabilistic map $Y()$ hides all functions of W with leakage ϵ if for every adversary $\mathcal{A}$, there exists an adversary $\mathcal{A}_*$ such that for all functions $f : \{0,1\}^* \to \{0,1\}^*$,*

$$\left|\Pr[\mathcal{A}(Y(W)) = f(W)] - \Pr[\mathcal{A}_*() = f(W)]\right| \leq \epsilon.$$

The map $Y()$ is called (t,ϵ)-entropically secure if $Y()$ hides all functions of W, for all t-sources W.

The definitions above make sense for any distance function on the strings held by Alice and Bob. In this paper, we only discuss Hamming distance over a vector space $\mathcal{F}_q^n$. Two cases will be of special interest: binary Hamming distance ($q = 2$), since it is the most widely studied, and $q \geq n$, since we will then be able to obtain information-theoretically optimal constructions.

[1] Min-entropy measures the difficulty of guessing W a priori: $\mathbf{H}_\infty(W) = -\log(\max_w \Pr[W = w])$.

1.1 Our Contributions

Our main result is the construction of entropically secure sketches for Hamming distance. We state the results for two settings: the binary alphabet ($q = 2$), and a large alphabet ($q \geq n$).

THEOREM 1. **(Binary Alphabet)** *There exist efficient (t, t', τ)-secure sketches for inputs in $\{0,1\}^n$ (with binary Hamming distance) which are also (t, ϵ)-entropically secure, such that, for infinitely many n,*

(1) *the tolerated error τ and residual entropy t' are $\Omega(n)$;*
(2) *the information leakage ϵ is exponentially small ($2^{-\Omega(n)}$) whenever the original min-entropy t is linear in n. (That is, whenever $t = \Omega(n)$ then we can find schemes where τ, t' and $\log\left(\frac{1}{\epsilon}\right)$ are $\Omega(n)$).*

(Large Alphabet) *If $q > n$ and $t > 2\tau \log(q)$, there exist efficient (t, t', τ, ϵ) entropically secure sketches over $\mathcal{F}_q^n$ such that $t' = t - 2\tau \log(q)$ and $\epsilon = O(2^{-t'/2})$.*

Before proceeding, a word about parameters: the original entropy t of the input W is given by the context in which W arises. The error tolerance τ will also typically be specified externally—it is the amount of noise to which W will likely be subject. Thus, the goal is to get both the (entropic) security $\log\left(\frac{1}{\epsilon}\right)$ and the residual min-entropy t' as high as possible. The quantity $\log\left(\frac{1}{\epsilon}\right)$ measures the difficulty of learning some function of W, while t' measures the difficulty of guessing W exactly. In particular, t' is bounded below by $\log\left(\frac{1}{\epsilon}\right)$ (roughly), since by the definition of entropic security the adversary's probability of predicting the identity function $f(W) = W$ is at most $\epsilon + 2^{-t} \approx \epsilon$. Thus, it is sufficient to look for sketches will tolerate τ errors and are (t, ϵ)-entropically secure for $\tau, \log\left(\frac{1}{\epsilon}\right) = \Omega(n)$. Theorem 1 states that such secure sketches do indeed exist.

In fact, for a large class of natural schemes—those where the sketch is actually a strong randomness extractor—we prove that $t' \geq 2\log\left(\frac{1}{\epsilon}\right) - \Theta(1)$; some of our constructions achieve this bound. For large enough alphabets, our constructions achieve both optimal leakage $\epsilon = 2^{-t'/2}$ and optimal min-entropy loss of $2\tau \log q$.

THE RELATION TO RANDOMNESS EXTRACTION. The starting point of the constructions is a result from earlier work stating that *randomness extractors* [23] are entropically secure, that is the output hides all functions of the source. We say a (randomized) map $Y()$ is (t, ϵ)-indistinguishable if for all pairs of t-sources W_1, W_2, the distributions $Y(W_1)$ and $Y(W_2)$ are ϵ-close. ($Y()$ is a randomness extractor in the special case where the output distribution is always close to uniform.) We will use the following result several times:

FACT 2 ([12], THM 2.1). *If $Y()$ is (t, ϵ)-entropically secure, then it is $(t-1, 4\epsilon)$-indistinguishable. Conversely, if $Y()$ is (t, ϵ)-indistinguishable, then it is $(t+2, 8\epsilon)$-entropically secure.*

The second implication is the more interesting of the two. In particular, our main result is really a construction of randomness extractors (which are indistinguishable) whose output can be used to correct errors in the input. They are *strong* randomness extractors in the sense of Nisan and Zuckerman [23]: all the random coins used by the extractor (the "seed") appear explicitly in the output. The construction is explained in Section 2; here we rephrase the main result in terms of extractors:

THEOREM 3. **(Binary alphabet)** *For any constant entropy rate t/n, there is an explicitly constructible ensemble of strong (t, ϵ)-extractors $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^p \times \{0,1\}^d$ with seed length $d = n$ such that (1) $\mathsf{Ext}()$ extracts* at most *a constant fraction $p = t(1 - \Omega(1))$ of entropy from the input with exponentially small error and (2) $\mathsf{Ext}()$ corrects a linear number τ of binary Hamming errors in the source. That is, there is a polynomial time algorithm Rec such that for any strings w, w' at distance at most τ, $\mathsf{Rec}(w', \mathsf{Ext}(w; R)) = w$ with probability 1.*

(Large alphabet) *For Hamming distance over $\mathcal{F}_q^n$, $q > n$, for any t, τ and n such that $t > 2\tau \log q$, there is an explicitly constructible strong (t, ϵ)-extractor which can correct τ errors efficiently, and which extracts $2\tau \log(q)$ bits with error satisfying $\log\left(\frac{1}{\epsilon}\right) \geq t/2 - \tau \log(q) - O(1)$.*

The previous theorem may initially appear strange when seen as a result on randomness extractors. Typically, the goal of extractor constructions is to *maximize* the length of the extracted string, and minimize the seed length d (for given min-entropy and error).

In our constructions however, the output length p of the extractor is the entropy loss of the corresponding secure sketch: because the output is random, every bit of output really decreases the entropy of the input by a bit. Thus, our goal is to *minimize* the output length while keeping the error (leakage) ϵ as low as possible and correcting as many errors τ as possible. To add to the confusion, the term *entropy loss* is overloaded here. In the context of extractors, the entropy loss typically refers to the disparity between the initial entropy and the number of extracted bits, i.e. $t - p$. In the context of secure sketches, this quantity is actually the *residual entropy* t', which we are trying to maximize. Thus, although it is helpful to think of our result in terms of extractors, one must bear in mind that the focus is on functionality—that is, error-tolerance—and not on maximizing the output.

Finally, one may also think of this result as adding to the list of connections between explicit constructions of different combinatorial objects—in this case, extractors and error-correcting codes. In the past, error-correcting codes have been used to construct extractors and vice-versa (see, e.g. [25]). However, this paper describes objects which are in some sense *both*. Fuzzy extractors [11] provide a different example of objects which combine the two requirements. The connections to fuzzy extractors are explained in Section 3.

APPLICATIONS. We present three applications of our results.

- *Key Re-Use in the Bounded Storage Model.* This is perhaps the least expected application of our technique, resolving the main open question left by Ding [10]. Ding considered the question of correcting errors in Maurer's bounded storage model [21]. In this model, Alice, Bob and the adversary all have temporary access to a huge, random string, X, but have very bounded memories (only enough to remember a fraction of the length of the string). Alice and Bob, using only a short, shared key, can derive a much longer shared key about which the adversary has no information, without making computational assumptions such as the existence of a pseudo-random generator. The model has received a lot of attention recently (see [10, 27, 18] and references therein), in particular because of a feature dubbed *everlasting security*: the same long term key can be re-used many times, and the session keys remain secure even if the adversary learns the long-term

key. One of the aspects limiting the usability of the current solutions comes from the fact that Alice and Bob must see (almost) exactly the same string X in order to ensure that they derive the same key: current protocols can only tolerate a negligibly small probability of error in each bit of X. By having Alice send a single message to Bob, Ding [10] showed that one can actually tolerate a constant error rate in X, at the expense of considerably weakening the key re-use property: the parties must synchronously and periodically update their long-term secret keys. We resolve this issue by showing that nearly optimal error-correction is possible *without sacrificing key re-use.*

- *Obfuscation and Perfectly One-Way Functions.* An obfuscator for a program P generates a scrambled circuit $\tilde{P}$ which allows one to evaluate P on any input, but leaks no additional information. Although code obfuscation is not possible in general (Barak *et al.* [1]), there are a few results showing that obfuscation of certain functions is possible in variants of the basic model, e.g. [19, 28]. Perfectly one-way hash functions, constructed by Canetti, Micciancio and Reingold [7], can be interpreted as obfuscating equality queries (which accept the input if and only if it is equal to some particular value w.), provided w has high enough min-entropy from the adversary's point of view. It is natural to ask whether it is possible to obfuscate more general proximity queries, which also accept inputs sufficiently close to w. This was explicitly mentioned as an open problem in [19], who observed that random oracles seem of little help for correcting unknown errors. We settle the problem in the affirmative under the assumption that w has high entropy (as in [7]). To do so, we construct error-tolerant perfectly one-way hash functions. Along the way, we simplify and improve the analysis of the noise-free construction of [7].
- *Stronger Privacy for Biometric Applications. Fuzzy extractors* were recently introduced [11] for extracting a secret, random key from noisy, non-uniform data such as biometric measurements. On input W, a fuzzy extractor outputs some public data P and a key $R(W)$, such that P can be used to recover R from subsequent readings of W, despite noise. In the constructions of [11], the public information P actually leaks some potentially sensitive information about the biometric W. Our results here are two-fold. On the one hand, we show that P *must* indeed leak some non-trivial amount of Shannon information about W. This conclusion is harder to prove for fuzzy extractors than for sketches, and relies on the geometry of $\{0,1\}^n$. On the other hand, we construct fuzzy extractors which leak no function (such as a sensitive substring) of the input W. This once again shows that Shannon security is stronger than entropic security.

This Abstract. Section 2 describes the construction of secure sketches which leak no partial information. Section 3, Section 4 and Section 5 describe the applications to private fuzzy extractors, to error-correction and key reuse in the bounded storage model and to resilient perfectly one-way hash functions, respectively. Due to space limitations, most proofs are deferred to the full version.

2. SKETCHES THAT HIDE ALL PARTIAL INFORMATION

This section describes the main technical construction of the paper (Theorem 1). Unless explicitly stated otherwise, most of the discussion below will concentrate on the more challenging case of binary alphabets. Our discussion refers often to the "code-offset" construction [2, 15]: if we view an error-correcting code as a function $C : \{0,1\}^k \to \{0,1\}^n$ with minimum distance d, the randomized map

$$S(w; R) = w \oplus C(R) \tag{1}$$

has entropy loss $t - t' = n - k$ [11]. It can correct $\tau = \lfloor (d-1)/2 \rfloor$ errors, and is efficient if and only if C has efficient encoding and error-correction algorithms. In the case of linear codes, this construction reduces to the syndrome construction in the introduction, since $w \oplus C(R)$ is a random element of the coset $\{x \in \{0,1\}^n \ : \ \mathsf{syn}_C(x) = \mathsf{syn}_C(w)\}$.

2.1 Overview of the Construction

Our starting point is the following fact about "small-bias" subsets of $\{0,1\}^n$ (defined below). If A is randomly drawn from a subset of sufficiently small "bias," and B is any random variable with sufficient min-entropy, then $A \oplus B$ is close to uniform on $\{0,1\}^n$. (This fact was also used to construct a entropically secure encryption scheme [24].) The intuition behind our approach, then, is simple:

If C itself is a small-bias set, then the code-offset construction $S(W) = W \oplus C(R)$ always yields distributions close to uniform, and hence $S()$ is entropically secure.

The problem with this intuition is that explicit constructions of codes with small bias are not known (in particular, such codes cannot be linear, and most explicitly constructible codes are linear). We circumvent this difficulty and construct explicit, *efficient* entropically secure sketches. We show that the code-offset construction can be made indistinguishable (even with linear codes) when the choice of error-correcting code is *randomized* as opposed to always using the same fixed code.

Suppose that we have a family of k-dimensional linear error-correcting codes $\{C_i\}_{i \in I}$ indexed by some set I. Consider sketches of the form

$$\begin{aligned} S(w; i) &= (i, \mathsf{syn}_{C_i}(w)) \ , \text{ for } \ i \leftarrow I, \text{ or, equivalently,} \\ S(w; i, x) &= (i, w \oplus C_i(x)) \ , \text{ for } \ i \leftarrow I, x \leftarrow \{0,1\}^k \end{aligned} \tag{2}$$

Below, we establish a necessary condition on the code family for the construction to leak no partial information about w.

1. We define a notion of "bias" for *families* of codes, and show that a small-bias family of codes also leads to an entropically secure sketch. This allows us to work with linear codes.
2. To illustrate the framework, we show that random linear codes are optimal in terms of both error-correction and entropic security (this corresponds to reproving the "left-over hash" lemma [14]).
3. We construct explicit, efficiently decodable, small-bias families of codes by considering a subset of binary images of a fixed code over a large (but constant-size) alphabet $GF(2^e)$.

A number of interesting observations come out of our analysis. First of all, we derive a general sufficient condition for

a set of *linear* functions to form a good randomness extractor; this may be of independent interest. We also obtain new bounds on the average weight enumerators of "generalized" algebraic-geometric codes.

Bias and Secrecy. The *bias* of a random variable A over $\{0,1\}^n$ is a (weak) measure of "pseudo-randomness": it measures how close A is to fooling all statistical tests that look only at the parity of a subset of bits. Formally, the bias of A with respect to a non-zero vector α is the distance between the dot product of α and A from a fair coin flip, that is

$$\mathsf{bias}_\alpha(A) \stackrel{\text{def}}{=} \mathbb{E}\left[(-1)^{\alpha \odot A}\right] = 2\Pr[\alpha \odot A = 1] - 1$$

The random variable A has bias δ if $|\mathsf{bias}_\alpha(A)| < \delta$ for all non-zero vectors $\alpha \in \{0,1\}^n$. The bias of a set C is the bias of the uniform distribution over that set. It is known that the map $Y(W;A) = W \oplus A$ is a (t,ϵ)-extractor whenever the bias of C is sufficiently small ($\delta \leq \epsilon 2^{-(n-t-1)/2}$), e.g. [3].

We generalize this to a family of sets by requiring that on average, the *squared* bias with respect to every α be low:

Definition 3. *A* family *of random variables (or sets)* $\{A_i\}_{i\in I}$ *is* δ*-biased if, for all* $\alpha \neq 0^n$, $\sqrt{\mathop{\mathbb{E}}_{i\leftarrow I}[\mathsf{bias}_\alpha(A_i)^2]} \leq \delta$.

Note that this is *not* equivalent, in general, to requiring that the expected bias be less than δ. There are two important cases:

Case 1. If C is a δ-biased set, then $\{C\}$ is a δ-biased set family with a single member.

Constructing codes with good minimum distance and negligible bias seems difficult. Such codes do exist: a completely random set C of 2^k elements will have both (1) minimum distance d, where $k/n \approx (1 - h_2(d/n))/2$ [20] and (2) bias approximately $2^{-(k-\log n)/2}$ [22]. However, these codes are neither explicitly constructed nor efficiently decodable. This raises a natural question:

Does there exist an explicitly-constructible ensemble of good codes with small bias and poly-time encoding and decoding algorithms (ideally, codes with linear rate and minimum distance, and exponentially small bias)?

To the best of our knowledge, the problem remains open.

Case 2. A family of linear codes $\{C_i\}_{i\in I}$ is δ*-biased* if there is no word which is often in the dual $C_i^\perp$ of a random code C_i from the family. Specifically, the bias of a linear space with respect to a vector α is always either 0 or 1:

$$\mathsf{bias}_\alpha(C_i) = \begin{cases} 0 & \text{if} \quad \alpha \notin C_i^\perp \\ 1 & \text{if} \quad \alpha \in C_i^\perp \end{cases}$$

Hence a family of codes is δ-biased if and only if $\Pr_{i\leftarrow I}[\alpha \in C_i^\perp] \leq \delta^2$, for every $\alpha \neq 0^n$.

Note that for a family of linear codes to satisfy Definition 3 the expected bias must be at most δ^2, while for a single set the bias need only be δ.

The general lemma below will allow us to prove that the randomized code-offset construction is indistinguishable (and hence entropically secure).

Lemma 4 (Small-Bias Families Yield Extractors). *Let* $\{A_i\}_{i\in I}$ *be a* δ*-biased family of random variables over* $\{0,1\}^n$*, with* $\delta \leq \epsilon \cdot 2^{-\frac{n-t-1}{2}}$*. For any* t*-source* B *(independent of* A_i*) the pair* $(I, A_I \oplus B)$ *is* ϵ*-close to uniform.*

The proof of Lemma 4 is a generalization of the proof that random walks on the hypercube converge quickly when the edge set is given by a small bias set. The basic idea is to bound the Fourier coefficients (over $\mathbb{Z}_2^n$) of the output distribution in order to show that it is close to uniform in the ℓ_2 norm.

In order to apply Lemma 4 we will need a family of error-correcting codes with small bias. Our construction is described in the next section, and summarized here:

Lemma 5 (Good Code Families Construction). *For any constant* $0 < \lambda < 1$*, there exists an explicitly constructible ensemble of (binary) linear code families which* efficiently *correct* $\tau = \Omega(n)$ *errors and have square bias* $\delta^2 < 2^{-\lambda n}$.

For any $q > n$ *and* $0 \leq \tau < n/2$*, there exists an explicitly constructible family of linear codes over* $\mathcal{F}_q^n$ *which efficiently corrects* τ *errors with dimension* $n - 2\tau$ *and square bias* $\delta^2 < O(q^{-(n-2\tau)})$.

Proof of **Theorem 1**. We can combine this lemma and Lemma 4 to prove our main result, i.e. that there are efficient, entropically secure sketches. For the binary case: If t/n is constant, we can set $\lambda = 1 - \frac{t}{2n}$. Picking a sequence of code families as in Lemma 5, we obtain a secure sketch scheme which corrects $\tau = \Omega(n)$ errors efficiently and is (t,ϵ)-entropically secure, where $\epsilon = \delta \cdot 2^{(n-t)/2+O(1)}$. Since $\delta^2 \leq 2^{-\lambda n} = 2^{t/2-n}$, the leakage ϵ is exponentially small.

To prove the statement large alphabets, fix n, q, t, and τ. In order to apply Lemma 4, we must first observe that the same statement holds over larger alphabets $\mathcal{F}_q$, with a similar bound $\epsilon = \delta \cdot 2^{\frac{(n\log(q)-t-1)}{2}}$. By Lemma 5, there is a code family which gives a secure sketch scheme which corrects τ errors (in $\mathcal{F}_q^n$) and is (t,ϵ)-entropically secure, where $\epsilon = O(2^{-(t-2\tau\log(q))/2})$. The dimension of the code is $n - 2\tau$ and so the entropy loss of the sketch is $2\tau \log q$. □

2.2 Small-Bias Families of Linear Codes

Inefficient Construction: Random Linear Codes. An easy observation is that the family of *all* linear codes of a particular dimension k has squared bias $\delta^2 = 2^{-k}$, although the codes are not known to be efficiently decodable. This bias is easily seen to be optimal. Random linear codes also exhibit the best known tradeoff between rate and distance for binary codes, as they lie near the Gilbert-Varshamov bound with high probability [20]. This gives us a point of reference with which to measure other constructions.

Efficient Construction via Random Binary Images. The basic idea behind our construction is to start from a single, fixed code C' over a large (but constant) alphabet, and consider a family of binary codes obtained by converting field elements to bit strings in different ways.

Let $\mathcal{F} = GF(q)$, where $q = 2^e$. Starting from a $[n', k', d]_q$ code C' over $\mathcal{F}$, we can construct a binary code by taking the *binary image* of C', that is by writing down the codewords of C' using some particular e-bit binary representation for elements of $\mathcal{F}$. More formally, fix any basis of the field $\mathcal{F}$ over $\mathbb{Z}_2$, and let $\mathsf{bin}(a) \in \{0,1\}^e$ be the binary representation of a field element a in the basis. For a vector $\alpha = (a_1, ..., a_{n'}) \in \mathcal{F}^{n'}$, let $\mathsf{bin}(\alpha)$ be the concatenation $(\mathsf{bin}(a_1), ..., \mathsf{bin}(a_{n'}))$. Finally, let $\mathsf{bin}(C')$ denote the set of binary images of the codewords, $\mathsf{bin}(C') \stackrel{\text{def}}{=} \{\mathsf{bin}(c) : \ c \in C'\}$.

We can randomize the code C' by multiplying each coordinate of the code by some random non-zero scalar in $\mathcal{F}$, and then taking the binary image of the result. Clearly, these operations affect neither the dimension nor the decodability of C': they are invertible and preserve Hamming distances in $\mathcal{F}^{n'}$. Describing the particular operations that were applied to the code requires $O(n' \log(q-1))$ bits to describe n' non-zero scalars.

When the initial code C' is a Reed-Solomon (RS) code or an algebraic-geometric (AG) code, the codes obtained as above are called *generalized* RS (resp. AG) codes. We prove a new bound the average square bias of these codes based on the minimum distance of the dual code of C'.

In the statement below, we consider general p-ary images instead of only binary ones. That is, when $q = p^e$ for a prime power e, we obtain the p-ary image of an element by writing it out in some fixed basis of $GF(q)$ over $GF(p)$.

LEMMA 6 (RANDOM p-ARY IMAGES). *Let C' be a linear $[n', k', d]_q$ code over $\mathcal{F} = GF(q)$, with $q = p^e$. Let $\{C'_i\}$ be the set of $[n', k', d]_q$ codes over $\mathcal{F}$ obtained by multiplying each coordinate by a non-zero scalar in $\mathcal{F}$. Let $C_i = \mathsf{bin}(C'_i)$ be the corresponding p-ary images. Then*

1. *The C_i are $[n, k, d]_p$ codes with $n = n'e$ and $k = k'e$. (Note that the rates k/n and k'/n' are equal).*
2. *If C' can correct τ errors in $GF(q)^{n'}$ efficiently, then each C_i can efficiently correct τ errors in $GF(p)^n$.*
3. *If $(C')^\perp$ has minimum distance $d_\perp$, then the average square bias of $\{C_i\}$ is*

$$\delta^2 = \max_{\alpha \in GF(p)^n, \alpha \neq 0^n} \left\{ \Pr_i[\alpha \in C_i^\perp] \right\} \leq 1/(q-1)^{d_\perp - 1}.$$

Note that in the last statement, the dual code $(C')^\perp$ is taken with respect to the dot product in $\mathcal{F}^{n'}$, while the dual code $C_i^\perp$ is taken with respect to the dot product in $GF(p)^n$.

USING REED-SOLOMON CODES. Recall, RS codes are a class of efficiently-decodable $[n', k', d]_q$ linear codes over a large alphabet $GF(q)$, $q \geq n'$, having distance $d = n' - k' + 1$ and dual distance $d_\perp = k' + 1$. Applying Lemma 6, we get $\delta^2 = (q-1)^{-d_\perp + 1} = (q-1)^{-k'} < 3q^{-k'}$. When q is in fact a power of 2 we get that binary images of RS codes have optimal bias $\log\left(\frac{1}{\delta}\right) = k/2 - O(1)$, just like random linear codes. Unfortunately, the conversion to a binary alphabet increases the code length and dimension without increasing the distance. Thus, these codes are only guaranteed to correct about $(n-k)/2e \leq (n-k)/2\log n$ binary errors.

Nevertheless, for *large alphabets*, these codes do very well. Specifically, setting $p = q$ and $e = 1$ (i.e., $n = n'$, $k = k'$), we get both optimal square bias $O(q^{-k'})$ and optimal distance $d = n - k + 1$. This gives the optimal large alphabet bound claimed in Lemma 5.

BINARY CODES VIA AG CODES. To overcome the distance reduction we experienced with RS codes, we use AG codes instead. Applying Lemma 6 to such codes yields the following lemma (whose proof is given in the full version), which implies Lemma 5.

LEMMA 7 (GOOD BINARY CODE FAMILIES). *For any constant $0 < R < 1$, and any $q = 2^{2i}$ where i is an integer, $i \geq 2$, there exists an explicitly constructible ensemble of binary code families which* efficiently *correct τ errors and have square bias δ^2 where:*

$$\tau \geq \frac{n}{\log q}\left(1 - R - \frac{1}{\sqrt{q}-1}\right) \quad \text{and}$$
$$\log\left(\tfrac{1}{\delta}\right) \geq \frac{n}{2}\left(R - \frac{1}{\sqrt{q}-1}\right)\left(1 - \frac{\log q}{q-1}\right)$$

3. SECRECY FOR FUZZY EXTRACTORS

Fuzzy extractors were introduced in [11] to cope with keys derived from biometrics and other noisy measurements.

DEFINITION 4 ([11]). *An $(t, \ell, \tau, \epsilon)$* fuzzy extractor *is a given by two efficient procedures* $(\mathsf{Gen}, \mathsf{Rep})$.

1. Gen *is a probabilistic generation procedure, which on input $w \in \mathcal{M}$ outputs an "extracted" string $R \in \{0,1\}^\ell$ and a public string P, such that for any t-source W, if $\langle R, P\rangle \leftarrow \mathsf{Gen}(W)$, then $\mathbf{SD}\left(\langle R, P\rangle, \langle U_\ell, P\rangle\right) \leq \epsilon$.*
2. Rep *is a deterministic reproduction procedure which allows one to recover R from the corresponding public string P and any vector w' close to w: for all $w, w' \in \mathcal{M}$ satisfying $\mathsf{dist}(w, w') \leq \tau$, if $\langle R, P\rangle \leftarrow \mathsf{Gen}(w)$, then we have $\mathsf{Rep}(w', P) = R$.*

We first explain the distinction between fuzzy extractors and the entropically secure sketches we constructed in Section 2, which also were extractors by themselves. The key difference is that in the case of entropically secure sketches one reconstructs W from W' and the extracted randomness $S(W)$. In the case of fuzzy extractors, one reconstructs W from W' and some "extra information" P, which is separate from the extracted randomness R. In particular, the value P could leak some non-trivial information about the input W. This is the case, for example, for the fuzzy extractors constructed in [11]. Thus, a natural question to ask in this context is whether it is possible to construct fuzzy extractors where the public value P does not leak any sensitive information about W; this is the subject of this section.

The first hope is to hide all information about W, that is to construct a fuzzy extractor where $P(W)$ and W are (close to) independent random variables. We show that this is impossible. For fuzzy extractors, as for secure sketches, leaking Shannon information is unavoidable even if the input distribution W is uniform (which is a valid t-source for any t). The proof of this result may be found in the full version.

THEOREM 8. *Assume $(\mathsf{Gen}, \mathsf{Rep})$ is a $(n, t, \ell, \tau, \epsilon)$ fuzzy extractor, and let P denote the public output of the generation algorithm $\mathsf{Gen}(W)$ on a uniform input W. Then as long as $\tau = \Omega(\sqrt{n})$, $\ell = \omega(1)$ and $\epsilon = o(1)$, then P reveals $\Omega(n)$ bits of information about W: $\mathbf{I}(W; P) = \Omega(n)$.*

Despite this limitation, one can achieve entropic security for fuzzy extractors—a weaker (but meaningful) notion of security.

DEFINITION 5. *A $(n, t, \ell, \tau, \epsilon)$ fuzzy extractor is called*

- *ϵ'-private if the pair $\langle R, P\rangle$ is (t, ϵ')-entropically secure.*
- uniform *if $\mathbf{SD}\left(\langle R, P\rangle, \langle U_\ell, U_{|P|}\rangle\right) \leq \epsilon$.*

A few observations are in place. First, recall that for secure sketches, we required that $S(W)$ be entropically secure. For fuzzy extractors, it is tempting to require only that

$P(W)$ be entropically secure since it is P which is published while R is supposed to be secret. However, we cannot guarantee that no information about R will be leaked in some higher level application (e.g., if R is used as a one-time pad to encrypt a known string). Requiring that the pair $\langle R, P\rangle$, be entropically secure protects against arbitrary information being revealed about both P and R. Second, by Fact 2 [12] we can see that a uniform fuzzy extractor is also $\epsilon' = 8\epsilon$-private for min-entropy $t + 2$. Thus, for our purposes it is sufficient to construct a uniform fuzzy extractor. Such an extractor can be viewed as a regular extractor Gen which extracts $\ell + |P|$ nearly random bits from W, but such that W is recoverable from "only" $|P|$ of these bits and any W' of distance at most τ from W. In this view, it is clear that we really want to minimize $|P|$ and maximize the length ℓ of the "actual" extracted randomness R. The fact that P is random is never used for the purposes of randomness extraction, but rather as a convenient tool to ensure that P leaks no sensitive information about W.

A Simple Construction. Recall that [11] made a simple observation that fuzzy extractors can be constructed from secure sketches by applying a strong randomness extractor[2] to the secure sketch. Specifically, if Ext is a strong randomness extractor using seed K, one sets $R = \mathsf{Ext}(W; K)$ and $P = \langle K, S(W)\rangle$. It is easy to see that this transformation preserves entropic security if we start with an entropically secure sketch S (here the indistinguishability view given by Fact 2 is very useful). In fact, the resulting private fuzzy extractor is actually uniform whenever our entropically secure sketch is an extractor (like in Theorem 3). In this case, the analysis of [11] and the near uniformity of our sketch implies that, for any t-source W, the joint distribution $\langle R, P\rangle = \langle \mathsf{Ext}(W; K), K, S(W)\rangle$ is within statistical distance $O(\epsilon)$ from a triple of uniform, independent random strings, which means that the resulting fuzzy sketch is uniform, and therefore private.

If $S(W)$ and Ext are strong extractors, this construction has an additional property: the value P contains all the randomness used in the fuzzy extractor. In other words, Gen by itself could be viewed as a strong randomness extractor which extracts a random string $\langle R, S(W)\rangle$. In this light, it is once again clear why we want to *minimize* the length of $S(W)$: as the total length of $S(W)$ and R can be at most t (in fact, $t - 2\log\left(\frac{1}{\epsilon}\right)$), minimizing $|S(W)|$ leaves more room for maximizing $|R| = \ell$. Slightly abusing the terminology, we will call such fuzzy extractors *strong*, by analogy with ordinary strong extractors (for which P is simply the seed).

Now, by using the entropically secure sketches constructed in Theorem 3 coupled with any strong extractor which extracts a constant fraction of the entropy (such as a pairwise-independent hash family), we get the following corollary.

Theorem 9. *For any constant entropy rate t/n, there exists an efficient construction of uniform strong $(n, t, \ell, \tau, \epsilon)$ fuzzy extractors such that (1) $\ell, \tau, \log\left(\frac{1}{\epsilon}\right)$ are all $\Omega(n)$; and (2) $|P|$ (and thus the seed which is part of P) is $O(n)$.*

These parameters are optimal up to a constant factor (where linear $|P|$ follows from Theorem 8). It would be interesting, however, to understand the precise optimal tradeoffs between various constant factors involved. Another interesting problem is to decrease the seed length of a strong uniform fuzzy extractor below $O(n)$.

4. NOISE TOLERANCE AND "EVERLASTING SECURITY"

In this section we resolve the main open question of [10]: we show that there is a noise-tolerant "locally computable extractor" which allows its key to be reused many times.

Bounded Storage Model (BSM). We first briefly recall the basics of the bounded storage model [21]. Alice and Bob share a short, "long-term" secret key K. A sequence of huge random strings $X_1, X_2, \ldots$[3] is broadcast to both of them. Alice and Bob then apply a deterministic function f_K to derive relatively short one-time pads $R_i = f_K(X_i)$. Traditionally, there are two main considerations in the bounded storage model: efficiency and *everlasting security*. Efficiency means that f_K depends on few bits of the source X_i (we will denote this number by n), and that these bits can be easily determined from the long-term key K alone. Security means that as long as the adversary does not know the secret key K and cannot store each source X_i in "its entirety", the one-time pads R_i are statistically close to uniform, even if the adversary later gets the long-term secret key K. A bit more formally (see [27] for a complete definition), we assume the adversary can store at most γN bits for some constant $\gamma < 1$, and can update its state $A_i \in \{0,1\}^{\gamma N}$ at time i as an arbitrary function of A_{i-1}, the current source X_i, as well as the past one-time pads $R_1, \ldots, R_{i-1}$. The claim now is that at the end of the game the joint distribution $\langle A_m, K, R_1, \ldots, R_m\rangle$ is ϵ-statistically close to $\langle A'_m, K, U_1, \ldots, U_m\rangle$, where U_i are independent, truly uniform keys of the same length as R_i, and A'_m is obtained in the same manner as A_m but using the U_i's in place of the R_i's.

The BSM has received a lot of attention recently (see [10, 27, 18] and references therein). The current technique for achieving everlasting security [18, 27] in this model is the "sample-then-extract" approach. The high-level idea is to have K consist of two keys K_s and K_e, where K_s is used to select a small portion X_s^i of the bits of X_i, and then K_e is used as a key for a strong randomness extractor [23]. Using optimal parameter settings, one can achieve a total key of size $O(\log N + \log\left(\frac{1}{\epsilon}\right))$.

Error-Correction in BSM. Recently Ding [10] considered the problem of the error-correction in the bounded storage model. Suppose that Bob does not necessarily receive the same string X_i as Alice, but instead receives some $\tilde{X}_i$ which is guaranteed (or expected) to be close to X_i in the Hamming distance. Ding proposed the following simple idea to overcome such errors, which we first describe for a single sample (i.e., $m = 1$). After receiving the source X and sampling the substring X^s (using K_s), Alice will simply send to Bob — over a public channel — the string $P = \mathsf{syn}_C(X^s)$, where C is a good error-correcting code.[4] Bob will sample

[2]Specifically, the extractor has to work for distributions of min-entropy at least $t' - \log\left(\frac{1}{\epsilon}\right)$, where t' is the residual min-entropy of the sketch. When the extractor comes from the leftover hash lemma, [11] also showed that the loss of $\log\left(\frac{1}{\epsilon}\right)$ above is unnecessary.

[3]More generally, it is sufficient that each X_i has high min-entropy conditioned on the other X_j for $j \neq i$.

[4]More precisely, if the adversary is allowed to corrupt δ-fraction of the bit in X, C should be able to correct slightly more than δ-fraction of errors.

the string $\tilde{X}^s$ which is going to be close to X^s (due to the properties of the sampler), which means that he can recover X^s from P and $\tilde{X}^s$, after which he can use K_e to extract the final randomness R. For any storage rate $\gamma < 1$, any (up to exponentially small) error ϵ and any number of sampled bits n, there exists an error rate $\delta > 0$ such that one can (a) extract $\Omega(n)$ bits which are ϵ-close to uniform, (b) tolerate δ-fraction of adversarial errors in the source, (c) still maintain optimal key size $O(\log N + \log\left(\frac{1}{\epsilon}\right))$, and (d) have Alice send $O(n)$ bits to Bob.

It is easy to see that this idea works for $t = 1$ and might appear to work for arbitrary number of repetitions t. However, Ding pointed out the following subtle problem. The value $\mathsf{syn}_C(X^s)$ leaks some information about X^s, which in turn could conceivably leak information about the long-term key K_s, since X_s depends on K_s. But the security in the BSM model crucially assumes that the sampling key K_s is close to *independent* from the source. Now, leaking P_1 conceivably leaks information about K_s, which in principle means that the attacker can store information A_2 which depends on K_s. Thus the conditional distribution of X_2 given the adversary's view can *no longer be argued to be independent from the sampling key* K_s, and the analysis does not go through.

Ding addressed this problem by making Alice and Bob synchronized and stateful. Specifically, after each communication they not only extract a fresh one-time pad R_i, but also refresh the *long-term key* K (specifically, K_s must be replaced). While Ding showed that this solution achieves very good parameters, it creates a lot of inconvenience for the sender and the receiver.

Our Construction. Using our technique, we resolve the main open problem of [10]: we achieve properties (a)-(d) above with a stateless scheme. We begin with a randomized scheme, and then derandomize it. The idea is to replace the fixed error-correcting code C by a code C_i chosen randomly from a family of codes $\{C_i\}$ we constructed in Section 2, which have the property that a syndrome of a randomly selected code is a randomness extractor. Namely, instead of sending $\mathsf{syn}_C(X^s)$ Alice will send a pair $\langle i, \mathsf{syn}_{C_i}(X^s)\rangle = S(X^s)$ to Bob (for random i). We also notice that since our current construction is randomized, Alice might as well sample a new (anyway very short) extraction key K_e for every time period, and then include K_e as part of the message transmitted to Bob. In this variant, Alice and Bob only share a single long-term sampling key K_s.

The concrete scheme above is a special case of the following more general paradigm, which is conceptually cleaner and easier to analyze. We will use the abstraction of uniform (or even just private) fuzzy extractors developed in Section 3. Recall, such extractor $\mathsf{Gen}(W)$ outputs a pair $\langle R, P\rangle$ which is statistically close to a pair of uniform random strings $\langle U_1, U_2\rangle$ (of the appropriate length), and such that R can be recovered from P and any W' which is reasonably close to W.

Suppose Alice and Bob share only a sampling key K_s of length $O(\log N + \log\left(\frac{1}{\epsilon}\right))$. Upon sampling of the substring X^s from X, Alice applies a uniform fuzzy extractor to produce $\langle R, P\rangle \leftarrow \mathsf{Gen}(X^s)$, views R as the current-period one-time pad, and sends P to Bob. Bob samples a different, but likely close string $\tilde{X}^s$, and then recovers R from $\tilde{X}^s$ and P. In the next time period, they repeat this process from scratch (using the same K_s).

It is easy to see that the randomized scheme above achieves the claimed parameters (a)-(d) when using the uniform fuzzy extractors from Theorem 9. We therefore only need to establish its security (below "high" denotes $\Omega(n)$, where n is the length of X^s). We start with one-time security. Recall, by the property of averaging samplers, proved in [27, 23], the joint distribution of $\langle A, K_s, X^s\rangle$ is statistically close to $\langle A, K_s, Y\rangle$, where $Y\ |_{K_s=a, g(X)=b}$ has high min-entropy, for every setting of a, b. Therefore, $\langle A, K_s, (R, P) = \mathsf{Gen}(X^s)\rangle$ is statistically close to $\langle A, K_s, \mathsf{Gen}(Y)\rangle$, which is in turn statistically close to $\langle A, K_s, (U_1, U_2)\rangle$, since Y has high entropy and Gen is a uniform fuzzy extractor. This immediately implies one-time security.

Now, given that the above "one-time" indistinguishability holds even *conditioned on the sampling key* K_s, the multiple time security of this locally computable extractor can be shown via a hybrid argument, much like for the error-free case (e.g., see [27]). (In contrast, using the original scheme of Ding one cannot necessarily condition on the sampling key K_s, since the syndrome could in fact reveal information about K_s.)

Finally, we notice that we can easily derandomize our construction, by increasing the secret key (on Alice's side only) by a constant factor. The idea is to first extract from our source X the $O(n)$ random bits Z needed for our randomized construction, and then to independently apply our randomized construction to X again, but now using Z. In order to extract Z, we can use any usual, "error-intolerant" extraction scheme with optimal key size $O(\log N + \log\left(\frac{1}{\epsilon}\right))$, such as the scheme of Vadhan [27]. Namely, in addition to sampling key K_s, Alice will store as part of her secret key another sampling key K'_s and an extraction key K'_e. Using $K' = (K'_s, K'_e)$, Alice first extracts $O(n)$ bits Z from X (which we know will look random to the attacker), and then uses our randomized scheme above with sampling key K_s and randomness Z. Notice, since Z will be leaked to the adversary (indeed, it will be sent in the clear to Bob), it does decrease the residual min-entropy of X. However, since $|Z|$ is only $O(n)$, this decrease is asymptotically negligible.

5. PERFECTLY ONE-WAY FUNCTIONS AND CODE OBFUSCATION

"Perfectly one-way" hash functions (POWFs) were introduced by Canetti [6] to formalize the common intuition that cryptographic hash functions reveal very little about their input. We adopt the somewhat simplified version of the definition used in the subsequent paper of Canetti, Micciancio and Reingold [7]; see [6, 7] for a discussion of the differences.

Informally, POWFs are *randomized* hash functions $w \mapsto H(w; R)$ which satisfy two properties. First, given w and y, one can verify that $y = H(w; r)$ for some value of the randomness r (that is, a computationally bounded adversary cannot produce a pair $w' \neq w$ which would pass the same test). Second, if R is random, then $H(w; R)$ reveals "no information" about w. The second requirement was formalized in [7] using entropic security. Our results apply in two different ways:

1. Noise Tolerance. We show how to construct "fuzzy"—that is, noise-resilient—perfect hash functions. The hash value for w allows one to verify whether a candidate string w' is close to w, but reveals nothing else about w.

This is a significant departure from the approach of Canetti

et al.. The motivation behind [6, 7] was to formalize the properties of an ideal "random oracle" which might be achievable by a real computer program. In contrast, even given a random oracle, it is not at all clear how to construct a *proximity* oracle for a particular value w (i.e. an oracle $B_{w,\tau}(\cdot)$ that accepts an input w' if and only if $\mathsf{dist}(w, w') \leq \tau$).

2. Improved Construction. We strengthen the results of [7] on information-theoretically-secure POWF's. First, we reduce the assumptions necessary for security: Canetti, Micciancio and Reingold [7] assume the existence of a collision-resistant hash function with an extra combinatorial property, *regularity* (balancedness), in order for their proof to go through. We modify the analysis so that the extra condition is unnecessary. Second, we improve the parameters of the [7] construction, roughly halving the requirement on the min-entropy of the input for the same level of security.

One- vs Many-Time Secrecy. The constructions we discuss are secure only when a bounded number of independent hash functions are revealed to an adversary. Canetti *et al.* also discussed *many-time* POWF constructions, which remain secure even when an unbounded number of independent hashes of an input w are revealed to a computationally bounded adversary. Extending our results to that setting remains an important open question.

The Relation to Code Obfuscation. Given a program P (say a Turing machine or a circuit), the goal of code obfuscation is to come up with a new program $\tilde{P}$ which computes the same functionality as P yet reveals nothing about its internal structure. The requirement can be formalized in various ways, but typically one asks that an adversary given $\tilde{P}$ learn no more, in some sense, than a "simulator" who is given only oracle access (i.e. input/output access) to a box which computes P.

Barak *et al.* [1] showed that general-purpose obfuscation is impossible: for some distributions on programs P and for any obfuscation technique which preserves the input/output behavior, there is a predicate $g()$ such that an adversary given $\tilde{P}$ will be able to predict $g(P)$ much better than an adversary given only oracle access to P. The proof is an elegant use of diagonalization. Drastically simplified, the idea is to run $\tilde{P}$ on its own description to compute $g(P)$.

In contrast, several results show that obfuscation is possible if one changes the model. Lynn *et al.* [19] showed that *point* functions can be obfuscated in the random oracle model. A point function $f_{w,a}()$ is given by two parameters, $w, a \in \{0,1\}^n$ such that $f_{w,a}(x) = a$ if $x = w$ and $\perp$ otherwise. The idea of [19] is, roughly, that storing $RO(w)$ allows one to check if $x = w$ but reveals nothing about w. The attack of Barak *et al.* fails since the "code" of the random oracle is not accessible in the model. Wee [28] recently showed that obfuscating point functions is possible even without the random oracle, at the cost of simulator whose running time depends on the quality of the simulation and a very strong complexity assumption. In a different vein, the perfectly one-way functions constructed by Canetti *et al.* [7] can be viewed as obfuscated point functions[5] (in the standard model), where the obfuscation holds as long as the adversary is sufficiently uncertain about the obfuscated value w to begin with.

These positive results deal only with point functions, and it is natural to ask whether or not one can obfuscate more general proximity queries, which accept inputs w' that are sufficiently close to w. This was explicitly mentioned as an open problem in [19].

The noise-resilient POWF's we construct here answer the question in the affirmative: they are weakly obfuscated versions of a proximity oracle where, as with [7], obfuscation holds as long as the target value w has enough entropy from the adversary's point of view.

5.1 Noise-resilient POWFs

Recall the two informal conditions on POWF's. Formalizing the first requirement is straightforward, though the hash function requires a key in order to get collision resistance against non-uniform adversaries. The second requirement is formalized using a definition almost identical to semantic security. Denote by R_n the space of random coins required by the hash, and by K_n the space of keys (for input lengths n). We say a family of keyed, randomized hash functions is a *one-time,* $(t(n), \tau(n), \epsilon(n))$ *noise-tolerant perfectly one-way hash function* if satisfies the following two definitions. An ordinary POWF corresponds to the special case $\tau(n) = 0$.

Definition 6 (Public Verifiability). *A ensemble of keyed randomized functions* $\mathcal{H} = \{H_k\}_{k \in K_n, n \in \mathbb{N}}$ *is* $\tau(n)$-publicly proximity-verifiable *if there is a polynomial-time verification algorithm* Ver *such that*

- *For all pairs* $w, w' \in \{0,1\}^n$ *such that* $\mathsf{dist}(w, w') \leq \tau(n)$*, we have* $\mathsf{Ver}(k, w, H_k(w; r)) = \text{ACC}$ *with probability 1 over* k *and* r*.*
- *For any PPT adversary* $\mathcal{A}$*, the probability over* k *and the coins of* $\mathcal{A}$ *that* $\mathcal{A}(k)$ *outputs a triple* $(w, \tilde{w}, c)$ *such that* $\mathsf{Ver}(k, w, c) = \mathsf{Ver}(k, \tilde{w}, c) = \text{ACC}$ *and* $\mathsf{dist}(w, \tilde{w}) > 2\tau(n)$ *is negligible in* n*. This probability is called the* soundness error*.*

Definition 7 (Semantic Security). *A ensemble of keyed randomized functions* $\mathcal{H} = \{H_k\}_{k \in K_n, n \in \mathbb{N}}$ *is* $(t(n), \tau(n), \epsilon(n))$*-semantically perfectly one-way if for every adversary* $\mathcal{A}$*, and for every ensemble* $\{W_n\}_{n \in \mathbb{N}}$ *of* t*-sources, there exists a circuit* $\mathcal{A}_*^{B_{w,\tau}(\cdot)}$*, which makes at most polynomially-many (in* n*) queries to the proximity oracle, such that for every function* $f : \{0,1\}^* \to \{0,1\}^*$ *and for every* n *and* $k \in K_n$*:*

$$\Pr_{\substack{w \leftarrow W_n \\ r \leftarrow R_n}}[\mathcal{A}(H_k(w; r)) = f(w)] - \Pr_{w \leftarrow W_n}[\mathcal{A}_*^{B_{w,\tau}(\cdot)} = f(w)] \leq \epsilon(n)$$

where $B_{w,\tau}(\cdot)$ *is the proximity oracle which accepts its input* w' *iff* $\mathsf{dist}(w, w') \leq \tau$*.*

The construction we give below actually satisfies a stronger definition, namely entropic security (Definition 2). That is, it satisfies the definition above even when the "simulator" $\mathcal{A}_*$ has no access at all to the proximity oracle. This is perhaps counter-intuitive, but it comes from the fact that the entropy of W is assumed to be high enough that the adversary can essentially never find an input on which the oracle accepts. This was also the case for the constructions of Canetti *et al.* [7] in the noise-free setting. The point is made explicitly in the analysis of Canetti *et al.* [7]. In

[5] In fact, Canetti *et al.* deal with identity queries, i.e. functions $Id_w(x)$ which are 1 if $x = w$ and 0 otherwise. It is easy to extend their results (and ours) to point functions, where a secret string a revealed when input is w.

our context, it is implicit in the fact that entropically secure sketches are possible for the assumed min-entropy.

CONSTRUCTION. The idea behind the main construction is simple: entropically secure sketches compose well with any ordinary POWF, as long as the residual entropy of the secret given the sketch is higher than the entropy requirement for the POWF. The proof of this lemma is given in the final version.

LEMMA 10. *Suppose that S is a $(n, t-1, t', \tau, \epsilon)$-entropically secure sketch, and $\{H_k\}$ is a family of ordinary POWF's (that is, $\tau = 0$) which is $(t' - \log\left(\frac{1}{\epsilon}\right)+1, \epsilon)$-perfectly one-way. Then the family $\{H'_k\}$ of randomized hash functions given by $\tilde{H}_k(w; r_1, r_2) \stackrel{def}{=} \langle S(w; r_1), H_k(w; r_2)\rangle$ is $(t+1, \tau, 2\epsilon)$-perfectly one-way.*

5.2 Improved Noise-free Construction

Before we can apply the generic construction of the previous section, we need to construct ordinary, non-noise-resilient POWF's. In the full version, we give a simpler and improved analysis of the construction of [7], in light of the equivalence between entropic security and indistinguishability. Specifically, the proposition below removes the regularity assumption on the collision-resistant family and roughly halves the sufficient min-entropy requirement t on input W.

PROPOSITION 11. *Suppose that*

- $\{\mathsf{crhf}_k(\cdot)\}_{k\in K_n, n\in\mathbb{N}}$ *is a collision-resistant hash family from n bits to $\ell = \ell(n)$ bits,*
- $\ell < t - 2\log\left(\frac{1}{\epsilon}\right)$*,*
- $\left\{\{\pi_i\}_{i\in\mathcal{I}}\right\}_{n\in\mathbb{N}}$ *is an ensemble of pairwise independent* permutations *of* $\{0,1\}^n$.

Then the ensemble of randomized hash functions given by: $H_k(w; i) = \langle i, \mathsf{crhf}_k(\pi_i(w))\rangle$ is (t, ϵ)-entropically secure.

To prove entropic security, it suffices to prove that the scheme is indistinguishable. The statement follows directly from a variant of the left-over hash lemma (Lemma 12 below), which states that combining pairwise-independent permutations with any arbitrary functions yields a "crooked" strong extractor: that is, the output may not look random, but it will look the same for all input distributions of sufficiently high entropy. Contrary to intuition, this statement does *not* follow directly from the left-over hash lemma (the hash functions here may be length-increasing and so the intermediate distribution $\langle I, h_I(X)\rangle$ is not close to uniform).

LEMMA 12. *Let $f : \{0,1\}^N \to \{0,1\}^\ell$ be an arbitrary function. If $\{h_i\}_{i\in\mathcal{I}}$ is a family of pairwise independent hash functions from n bits to N bits and X is a t-source (in $\{0,1\}^n$) with $t \geq \ell + 2\log\left(\frac{1}{\epsilon}\right) + 1$, then the distributions $\langle I, f(h_I(X))\rangle$ and $\langle I, f(U_N)\rangle$ are ϵ-far where $I \leftarrow \mathcal{I}$, $U_N \leftarrow \{0,1\}^N$ (both drawn independently of X).*

5.3 Final Construction

We can now combine Theorem 1 with Proposition 11 and Lemma 10, and obtain the following result:

THEOREM 13. *If collision-resistant hash functions exist, then for any initial entropy $t = \Omega(n)$, there exists a one-time (t, τ, ϵ) noise-tolerant POWF ensemble which tolerates a linear number of errors $\tau = \Omega(n)$, achieves exponential security $\epsilon = 2^{-\Omega(n)}$ and is publicly verifiable with negligible soundness error.*

6. REFERENCES

[1] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. Vadhan, K. Yang. On the (Im)possibility of Obfuscating Programs. In *Advances in Cryptology — CRYPTO 2001*, pp. 1–18.

[2] C. Bennett, G. Brassard, and J. Robert. Privacy Amplification by Public Discussion. *SIAM J. on Computing*, **17**(2), pp. 210–229, 1988.

[3] Eli Ben-Sasson, Madhu Sudan, Salil P. Vadhan, Avi Wigderson: Randomness-efficient low degree tests and short PCPs via epsilon-biased sets. STOC 2003: 612-621

[4] Gilles Brassard, Louis Salvail. Secret-Key Reconciliation by Public Discussion. In *Advances in Cryptology — EUROCRYPT 1993*, p. 410–423.

[5] Christian Cachin, Ueli M. Maurer. Linking Information Reconciliation and Privacy Amplification. In *J. Cryptology*, **10**(2), 97-110, 1997.

[6] R. Canetti. Towards realizing random oracles: Hash functions that hide all partial information. In *Advances in Cryptology — CRYPTO 1997*.

[7] R. Canetti, D. Micciancio, O. Reingold. Perfectly One-Way Probabilistic Hash Functions. In *Proc. 30th ACM Symp. on Theory of Computing*, 1998, pp. 131–140.

[8] V. Chauhan and A. Trachtenberg. Reconciliation puzzles. IEEE Globecom 2004.

[9] Graham Cormode, Mike Paterson, Süleyman Cenk Sahinalp, Uzi Vishkin. Communication complexity of document exchange. *Proc. ACM Symp. on Discrete Algorithms, 2000*, p. 197–206.

[10] Y.Z. Ding. Error Correction in the Bounded Storage Model. In *Theory of Cryptography 2005*.

[11] Y. Dodis, L. Reyzin and A. Smith. Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. In *Advances in Cryptology — EUROCRYPT 2004*.

[12] Y. Dodis and A. Smith. Entropic Security and the Encryption of High-Entropy Messages. In *Theory of Cryptography 2005*.

[13] S. Goldwasser and S. Micali. Probabilistic encryption. *JCSS*, **28**(2), pp. 270–299, April 1984.

[14] R. Impagliazzo and D. Zuckerman. How to Recycle Random Bits. In *Proc. 30th IEEE Symp. on Foundations of Computer Science*, 1989.

[15] A. Juels, M. Wattenberg. A Fuzzy Commitment Scheme. In *Proc. ACM Conf. Computer and Communications Security, 1999*, pp. 28–36.

[16] A. Juels and M. Sudan. A Fuzzy Vault Scheme. In *IEEE International Symposium on Information Theory*, 2002.

[17] Shengli Liu and Henk C. A. Van Tilborg and Marten Van Dijk. Practical Protocol for Advantage Distillation and Information Reconciliation. *Des. Codes Cryptography*, **30**(1), 39–62, 2003.

[18] Chi-Jen Lu. Encryption against Storage-Bounded Adversaries from On-Line Strong Extractors. *J. Cryptology*, 17(1): 27-42 (2004).

[19] Ben Lynn, Manoj Prabhakaran, Amit Sahai. Positive Results and Techniques for Obfuscation. *Advances in Cryptology — EUROCRYPT 2004*, p. 20–39.

[20] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error-Correcting Codes*, North-Holland, Amsterdam, New York, Oxford, 1978.

[21] U. Maurer. Secret Key Agreement by Public Discussion. *IEEE Trans. on Info. Theory*, 39(3):733–742, 1993.

[22] J. Naor, M. Naor. Small-Bias Probability Spaces: Efficient Constructions and Applications.*SIAM J. Comput.* 22(4): 838-856 (1993).

[23] N. Nisan, D. Zuckerman. Randomness is Linear in Space. In *JCSS*, **52**(1), pp. 43–52, 1996.

[24] A. Russell and Wang. How to Fool an Unbounded Adversary with a Short Key. In *Advances in Cryptology — EUROCRYPT 2002*.

[25] R. Shaltiel. Recent developments in Explicit Constructions of Extractors. *Bulletin of the EATCS*, **77**, pp. 67–95, 2002.

[26] A. Smith. Maintaining Secrecy When Information Leakage is Unavoidable. Ph.D. Thesis, Massachusetts Institute of Technology, 2004.

[27] Salil P. Vadhan. Constructing Locally Computable Extractors and Cryptosystems in the Bounded-Storage Model. *J. Cryptology* 17(1): 43-77 (2004).

[28] Hoeteck Wee. On Obfuscating Point Functions. (These Proceedings.) *Proc. 37th ACM Symp. on Theory of Computing*, 2005.