J. Cryptology (2004) 17: 27–42
DOI: 10.1007/s00145-003-0217-1

Journal of
CRYPTOLOGY

© 2003 International Association for Cryptologic Research

# Encryption against Storage-Bounded Adversaries from On-Line Strong Extractors*

Chi-Jen Lu
Institute of Information Science, Academia Sinica,
Taipei, Taiwan, Republic of China
cjlu@iis.sinica.edu.tw

Communicated by Oded Goldreich

Received 12 July 2002 and revised 16 December 2002
Online publication 6 August 2003

**Abstract.** We study the problem of information-theoretically secure encryption in the bounded-storage model introduced by Maurer [15]. The sole assumption of this model is a limited storage bound on an eavesdropper Eve, who is even allowed to be computationally unbounded. Suppose a sender Alice and a receiver Bob agreed on a short private key beforehand, and there is a long public random string accessible by all parties, say broadcast from a satellite or sent by Alice. Eve can only store some partial information of this long random string due to her limited storage. Alice and Bob read the public random string using the shared private key, and produce a one-time pad for encryption or decryption. In this setting, Aumann et al. [3] proposed protocols with a nice property called everlasting security, which says that the security holds even if Eve later manages to obtain that private key. Ding and Rabin [8] gave a better analysis showing that the same private key can be securely reused for an exponential number of times, against some adaptive attacks.

We show that an encryption scheme with such nice properties can be derived immediately from any strong randomness extractor, a function which extracts randomness from a slightly random source, so that its output and its seed together are almost random. To have an efficient encryption scheme, one needs a strong extractor that can be evaluated in an on-line and efficient way. We give one such construction, which yields an encryption scheme that has the nice security properties as before but now can encrypt longer messages using shorter private keys.

**Key words.** Bounded-storage model, Everlasting security, Randomness extractors, List-decodable codes, Expander graphs.

## 1. Introduction

Almost all cryptographic protocols in use today are based on some intractability assumptions. That is, adversaries are assumed to be computationally bounded, and some

* The conference version of this paper appeared in *CRYPTO* 2002.

problems are assumed to be computationally hard. However, no such complexity lower bound has been proved, and in fact it seems to remain far beyond the reach of current techniques in complexity theory. So it is possible that future advances in cryptanalysis or computer technology may jeopardize the security of today's cryptographic systems. One extreme then is to look for protocols with provable information-theoretical security, which are secure against computationally unbounded adversaries, but this is known to be impossible in general. Taking a step back, it would still be nice if one could prove information-theoretical security basing only on some minimal and reasonable assumption. Maurer proposed one such model, called the bounded-storage model [15], where the only assumption is that an adversary has a bounded amount of storage. To exploit such weakness of an adversary, a very long public random string, accessible by all parties, is usually employed. With respect to this model, several interesting cryptographic protocols have been proposed, with provable information-theoretical security.

One important task in cryptography is secure transmission against eavesdropping, where a sender Alice wants to send a message to a receiver Bob in a way to keep an eavesdropper Eve from learning the content. In this paper we study private-key encryption using one-time pads in the bounded-storage model. Assume Alice and Bob share a private key beforehand. Then a long public random string $X$ is generated, say broadcast from a satellite or sent by Alice, which is accessible by all parties. Eve has a limited storage so that only some partial information about $X$ can be stored. For the protocol to be efficient, Alice and Bob should require much less storage than the bound placed on Eve. Alice and Bob read the string $X$ on the fly, and compute a one-time pad $Z$. Then Alice encrypts her message as $C = M \oplus Z$ and sends $C$ to Bob. When $X$ is sent, Eve computes and stores some partial information about $X$, hoping later to recover $M$ after eavesdropping the cipher-text $C$. In this setting, Aumann et al. [3] gave protocols, improving those of Maurer [15] and Cachin and Maurer [6], which enjoy a nice provable property called *everlasting security*. This is an information-theoretical security property and guarantees secrecy for Alice and Bob even if Eve later after the transmission manages to obtain that private key. However, how would Alice and Bob share a private key in the first place? They might need to set up some truly secret channel for that, but this seems inconvenient in general and may even be infeasible on occasions. The private key can actually be sent via today's public-key encryption. The everlasting security guarantees that even if Eve later obtains that private key after the transmission of $X$, say by breaking the public-key encryption, the message $M$ will still remain information-theoretically secure. Such a feature seems quite attractive, as the security is guaranteed by the limitation of current storage technology, and will not be affected by future advances of any kind. This is possible as some crucial information has been lost forever. Shortly after, Ding and Rabin [8] gave a better analysis showing that the same private key can be securely reused for an exponential number of times in a way that each encryption remains secure even after revealing all previous plain-texts.

Let $k$ be the security parameter such that Eve cannot distinguish different messages with advantage more than $2^{-k}$. Let $m$ be the length of the message to be encrypted. Let $n$ be the length of the public random string $X$. Assume that the eavesdropper's storage bound is $\nu n$, for some constant $\nu < 1$. Given these parameters, we would like to find an encryption scheme that minimizes the length of the private key. Also, as the public

random string $X$ is typically very long and broadcast at a very high rate, the encryption scheme should be able to process $X$ in an on-line and efficient fashion.

Observe that as Eve can only store $\nu n$ bits of information $Q$ from the $n$-bit random string $X$, very likely a substantial amount of randomness still remains in $X$ relative to $Q$. The remaining randomness may be crude and not directly applicable, so one would like to have it purified. This is exactly the issue addressed in the research on constructing the so-called *extractors*, first explicitly defined by Nisan and Zuckerman [17]. Extractors turn out to have many important applications (see [16] for a survey) and have become a subject of intense study. An extractor is a function that given any source with enough randomness uses a short perfect random seed to extract many bits which are close to random. We can use the output from an extractor as a one-time pad for encryption as it looks random to Eve. A strong extractor is an extractor with a stronger guarantee that its output and its seed together are close to random, which gives the following nice properties. On one hand, with high probability, the output of the extractor still looks random even given the value of the seed, which guarantees the property of everlasting security of [3] discussed above. On the other hand, with high probability, the seed still looks random even given the output value of the extractor, which implies that the same private key can be reused again for the next encryption even if Eve knows the pads of previous encryptions. Formally, we show that any strong extractor immediately yields an encryption scheme with everlasting security, against an exponential number of adaptive attacks, the same property enjoyed by protocols of [3] and [8].

However, not every strong extractor is suitable for an encryption scheme in the setting we consider here. As the public random string is broadcast at a very high rate and each party does not have enough memory to store the whole string, we need a strong extractor that can be evaluated in an on-line and very efficient way. One way of constructing extractors, introduced by Trevisan [19], is to encode the input string using some list-decodable code and then project that codeword onto some dimensions determined by the random seed as output. A list-decodable code, roughly speaking, is a code guaranteeing some upper bound on the number of codewords in any Hamming ball of certain radius. For the security parameter $k$ so that Eve's distinguishing probability is at most $2^{-k}$, we need the number of codewords to be at most $2^{\mu n+O(k)}$, for some constant $\mu \in (0, 1)$, in any ball of relative radius $\frac{1}{2}-2^{-O(k)}$. The work of Aumann et al. [3], [8] can be understood within this framework. Their main technical contribution can be seen as constructing such a list-decodable code mapping an $n$-bit string to an $n^{O(k)}$-bit codeword, where each bit of a codeword is the parity of $O(k)$ input bits. For an extractor with one output bit, they encode the input, pick a random dimension of that codeword, and output the bit there, needing a seed of length $O(k \log n)$ for sampling. To output $m$ bits, they independently choose $m$ random dimensions, and project that codeword onto those $m$ dimensions, needing a seed of length $O(mk \log n)$. So in their encryption scheme, a private key of length $O(mk \log n)$ is required to encode a message of length $m$. That is, in order to achieve the everlasting security, they need a private key much longer than the message, which may limit the applicability of their protocol. Our improvement comes from two directions. First, we construct such a list-decodable code mapping an $n$-bit string onto a codeword of length $n2^{O(k)}$, where each bit of a codeword is again the parity of $O(k)$ input bits, but now selected via a random walk on some expander graph. An essentially same construction appeared in [1] but was used for a different purpose: to amplify further

the distance of a code which already has some linear distance. Here we use this code construction alone and prove its list-decodability property, which seems new and may have interest of its own. Next, to have an $m$-bit output, if we simply project the codeword onto $m$ random dimensions, needing a seed of length $O(m(k+\log n))$, we already have improvement over that of [3] and [8]. For large $m$, we follow the approach of [19] and [18] by picking some pseudo-random collection of $m$ dimensions, instead of $m$ random dimensions, for projection. As a result, for any $m \leq n^{\gamma}$ with $\gamma \in (0, 1)$, we only need a seed of length $O((k+\log n)^2/\log n)$. Note that $n$ is typically much larger than $k$. When setting $n = 2^{\Omega(k)}$, we get an encryption scheme using a private key of length $O(\log n)$, which is a dramatic improvement over [3] and [8].

To encrypt an $m$-bit message, Alice and Bob need to prepare $m$ families of $O(k)$ indices and remember them. We use random walk on expander graphs and some set system to determine those $m$ families, which seems to cost some extra computation compared with [3] and [8]. However, this small extra effort is only required during the preprocessing phase, when computation time may not be a concern, so hopefully would not be an issue from a practical point of view. During the broadcast of the long public random string $X$, we do exactly the same as that of [3] and [8]. That is, Alice and Bob compute the $m$-bit one-time pad, where each of the $m$ bits is a parity of $O(k)$ bits from $X$, which can be done in an extremely fast way.

Our protocol enjoys the following key expansion property, which is addressed by a new result of Dziembowski and Maurer [9]. In [3], to generate a one-time pad of length $m$, Alice and Bob need first to agree on a private key of length $O(mk \log n)$, much longer than the pad they want to generate. In [8] a key doubling technique is used to generate a one-time pad of length $m$ using a private key of length $O(k \log n)$, but it requires a public random string of length $\Theta(nk \log m \log n)$, much longer than Eve's storage bound $vn$. So an open question was whether one could generate a pad longer than the private key while using a public random string only slightly longer than Eve's storage bound. The main contribution of Dziembowski and Maurer [9] is to settle this question, and they need a private key of length $O(k \log n)$. Independently, we also settle this question but via a very different method, needing a private key of length $O((k+\log n)^2/\log n)$, which is smaller than theirs for $k = O(\log^2 n)$. Both [9] and our result allow the ratio between the length of public random string and Eve's storage bound to be arbitrarily close to one.[1] On the other hand, we do not require the long public random string $X$ to be perfectly random, while this is not clear in [9]. In reality, perfect random sources may not be available and one may have to rely on sources of lower quality. Our scheme works as long as $X$ contains enough amount of randomness, which only needs to be slightly larger than Eve's storage bound.

In addition to obtaining a better construction, we try to place this line of research in an appropriate framework, where ideas could be understood in a more intuitive and perhaps deeper way and various powerful tools have been developed. Most of the techniques we use are standard in the research area on pseudo-randomness. The idea of using

[1] This is the case when we produce a single one-time pad for a single message. To produce several one-time pads for several messages, we need several blocks of public random strings. The ratio is still arbitrarily close to one if it is with respect to the length of each block, but it is not so with respect to the total length of these blocks.

extractors in such a setting actually appeared implicitly before in the work of Cachin and Maurer [6]. They first sample bits from $X$ in a pairwise independent way, and then apply universal hash functions on these bits. In retrospect, their approach does produce an on-line strong extractor. Since then the theory of pseudo-randomness has made some advancement, which enables us to have a better understanding and derive stronger results. Recently, Vadhan has found a new construction that reduces the private key length down to $O(k+\log n)$ [20], which beats ours when $k = \omega(\log n)$. His approach is similar to that of Cachin and Maurer [6], but he is able to utilize stronger tools developed afterwards. First, a good oblivious sampler is used to select a subset of positions beforehand, which will likely inherit enough min-entropy from $X$. When $X$ is broadcast, the bits in those positions are stored in real time, which takes care of the issue that $X$ comes at a very high rate. Finally, a good strong extractor is applied on those bits to extract randomness, and now one can afford to use any strong extractor with good parameters without needing to worry about how fast it can be evaluated. Another related result is that of Bar-Yossef et al. [4]. They show that the extractors of Trevisan [19] and Raz et al. [18] can be turned into on-line ones by using the concatenated Reed–Solomon and Hadamard code together with the weak design of Hartman and Raz [12]. The seed length is $O((k + \log n)^2/\log n)$, the same as ours. However, their concern was just to be able to read $X$ in one-pass with small space. In fact, the evaluation of their extractors is more involved and time-consuming, so they may not be suited for our setting here with $X$ coming at a very rapid pace.

Notation, definitions, and some simple facts are given in Section 2. In Section 3 we prove that any strong extractor gives an encryption scheme with everlasting security. In Section 4 we construct an efficient on-line strong extractor which yields an efficient encryption scheme.

## 2. Preliminaries

For a positive integer $n$, let $[n]$ denote the set $\{1, \ldots, n\}$. For an $n$-dimensional vector $m$, let $m(i)$, for $i \in [n]$, denote the component in the $i$th dimension of $m$, and let $m(I)$, for $I \subseteq [n]$, denote the vector consisting of those $m(i)$'s for $i \in I$. For our convenience, we use $\{-1, 1\}$, instead of the usual $\{0, 1\}$, for the binary values unless noted otherwise. For a positive integer $n$, let $U^n$ denote the uniform distribution over $\{-1, 1\}^n$, and we omit the superscript $n$ when no confusion is caused. All the logarithms in this paper have base 2.

The distance between two vectors $x, y$ is defined as $d_{\mathrm{H}}(x, y) \equiv |\{i : x(i) \neq y(i)\}|$, which is also known as their *Hamming* distance.

**Definition 1.** A mapping $C\colon \{-1, 1\}^n \to \{-1, 1\}^{\bar{n}}$ defines a code, with $\{C(x) : x \in \{-1, 1\}^n\}$ as the set of codewords. It is called a *list-decodable code* with parameter $(\frac{1}{2} - \varepsilon, L)$, or a $(\frac{1}{2} - \varepsilon, L)$-*list code*, if for any $z \in \{-1, 1\}^{\bar{n}}$, there are at most $L$ codewords within distance $(\frac{1}{2} - \varepsilon)\bar{n}$ from $z$.

We also need a way to measure how close two distributions are.

**Definition 2.** The distance between two distributions $A, B$ is defined as $\|A - B\| \equiv \frac{1}{2}\sum_x |\Pr[A = x] - \Pr[B = x]|$.

This is usually called the variational distance, and it is easy to verify that $\|A-B\| \leq 1$. We say that a distribution is $\varepsilon$-random if its distance to the uniform one is at most $\varepsilon$. Here is a simple but useful lemma, which is proved in Appendix A.

**Lemma 1.** *Suppose the distribution $A$ is independent of distributions $B$ and $B'$. Then for any function $g$, $\|\langle g(A, B), B\rangle - \langle g(A, B'), B'\rangle\| = \|B - B'\|$.*

We need a measure to quantify the randomness contained in a slightly random source. Shannon's entropy function captures the average randomness of a source, while we need some worst-case measure of randomness instead.

**Definition 3.** For a distribution $X$, its *min-entropy* is defined as $H_\infty(X) \equiv \min_x \log(1/\Pr[X = x])$.

So $H_\infty(X) \geq r$ if and only if $\Pr[X = x] \leq 2^{-r}$ for any $x$. For a distribution with enough randomness, guaranteed by its min-entropy, we look for a procedure that can purify the randomness by using a short random seed as a catalyst.

**Definition 4** [17]. A function $\text{EXT}$: $\{-1, 1\}^n \times \{-1, 1\}^s \to \{-1, 1\}^m$ is called a *strong $(r, \varepsilon)$-extractor* if for any distribution $X$ over $\{-1, 1\}^n$ with $H_\infty(X) \geq r$ and for $Y = U^s$, the distribution of $\langle \text{EXT}(X, Y), Y\rangle$ is $\varepsilon$-random.

The usual definition of extractors only requires the distribution of $\text{EXT}(X, Y)$ being $\varepsilon$-random. The stronger requirement of a strong extractor EXT gives the following nice property. With high probability, the distribution of $\text{EXT}(X, Y)$ still looks random even given the value of $Y$, and also with high probability, the distribution of $Y$ still looks random even given the value of $\text{EXT}(X, Y)$. This is guaranteed by the following lemma, which is proved in Appendix B.

**Lemma 2.** *Suppose the distribution $\langle A, B\rangle$ is $\varepsilon$-random. Then the probability over $b \in B$ that the distribution $\langle A \mid B = b\rangle$ is not $\sqrt{\varepsilon}$-random is at most $2\sqrt{\varepsilon}$.*

We consider private-key encryption schemes using one-time pads. In such a scheme, a sender Alice and a receiver Bob first agree on a private key $Y$ and then later generate a one-time pad $Z_Y$ so that the message $M$ is encrypted as $C = M \oplus Z_Y$, where $\oplus$ denote the bitwise exclusive or operation. Note that an eavesdropper Eve with $C$ knows $M$ if and only if she knows $Z_Y$. So for the security of such a scheme, it suffices to show that $Z_Y$ looks random to Eve, given her information $Q$. Aumann et al. [3] introduced the following stronger notion of security.

**Definition 5.** A private-key encryption scheme is said to have *everlasting security of degree $k$* if the probability over $y \in Y$ and $q \in Q$ that the distribution $\langle Z_Y \mid Y = y, Q = q\rangle$ is not $2^{-k}$-random is at most $2^{-\Omega(k)}$.

This guarantees that even if Eve is given that private key $Y$ afterwards, she is still unlikely to know anything about the message $M$.

## 3. Everlasting Security from Strong Extractors

In this section we show that using a strong extractor for encryption guarantees the everlasting security of [3] and [8]. Recall the following scenario. Before the transmission of an $m$-bit message $M$, a sender Alice and a receiver Bob agree on a random private key $Y$. Then they read the $n$-bit public random string $X$, compute the one-time pad $Z$, and the encrypted message $C = M \oplus Z$ is sent from Alice to Bob. An adversary Eve uses a function $f$ to store $\nu n$ bits of information $Q = f(X)$ from the public string $X$, and then eavesdrops the encrypted message $C$. If later Eve is given that private key $Y$, can she obtain any information about $M$?

For the security parameter $k$, which is usually much smaller than $n$, choose $\varepsilon = 2^{-2k}$. Pick $\mu < 1-\nu$, say $\mu = 0.99-\nu$, so that $2^{-(1-\mu-\nu)n} \leq \varepsilon/2$, and let $r = \mu n$ . We will use a strong $(r, \varepsilon/2)$-extractor $\text{EXT}$: $\{-1, 1\}^n \times \{-1, 1\}^s \to \{-1, 1\}^m$. The private key $Y$ is selected randomly according to the distribution $U^s$ and the one-time pad is generated as $Z = \text{EXT}(X, Y)$. For $M = C \oplus Z$, Eve knows $M$ if and only if she knows $Z$. Note that $X$ conditioned on $Q$ is likely to contain enough randomness, so very likely the distribution of $Z = \text{EXT}(X, Y)$ would look random to Eve. This idea actually appeared before in the setting of constructing pseudo-random generators for space-bounded computation [17]. Furthermore, with $\text{EXT}$ being a strong extractor, we show that $\text{EXT}(X, Y)$ still looks random even given the value of $Y$.

**Lemma 3.** $\|\langle f(X), Y, \text{EXT}(X, Y)\rangle - \langle f(X), U^s, U^m\rangle\| \leq \varepsilon$.

**Proof.** First, note that $\|\langle f(X), Y, \text{EXT}(X, Y)\rangle - \langle f(X), U^m, U^s\rangle\|$ equals

$$\sum_q \Pr[f(X) = q] \cdot \|\langle Y, \text{EXT}(X, Y) \mid f(X) = q\rangle - \langle U^m, U^s \mid f(X) = q\rangle\|.$$

We call a value $q \in \{-1, 1\}^{\nu n}$ *bad* if $\Pr[f(X) = q] \leq 2^{-(1-\mu)n}$, and let $B$ denote this set of bad $q$'s. For $q \notin B$, $\Pr[X = x \mid f(X) = q] \leq 2^{-n}/2^{-(1-\mu)n} = 2^{-\mu n}$ for any $x$, so $H_\infty(X \mid f(X) = q) \geq \mu n$. Then for $q \notin B$,

$$\begin{aligned}
&\|\langle Y, \text{EXT}(X, Y) \mid f(X) = q\rangle - \langle U^s, U^m \mid f(X) = q\rangle\| \\
&\qquad = \|\langle Y, \text{EXT}(\langle X \mid f(X) = q\rangle, Y)\rangle - \langle U^s, U^m\rangle\| \\
&\qquad \leq \frac{\varepsilon}{2}.
\end{aligned}$$

For $q \in B$, we only have $\|\langle \text{EXT}(X, Y), Y \mid f(X) = q\rangle - \langle U^m, U^s \mid f(X) = q\rangle\| \leq 1$, but fortunately it is unlikely to have a bad $q$ as

$$\begin{aligned}
\sum_{q \in B} \Pr[f(X) = q] &\leq \sum_{q \in B} 2^{-(1-\mu)n} \\
&\leq 2^{(\nu-1+\mu)n} \\
&\leq \frac{\varepsilon}{2}.
\end{aligned}$$

Thus,

$$\begin{aligned}
&\|\langle f(X), Y, \text{EXT}(X,Y)\rangle - \langle f(X), U^s, U^m\rangle\| \\
&\qquad = \sum_q \Pr[f(X)=q] \\
&\qquad\quad \cdot \|\langle Y, \text{EXT}(X,Y) \mid f(X)=q\rangle - \langle U^s, U^m \mid f(X)=q\rangle\| \\
&\qquad \le \sum_{q\in B} \Pr[f(X)=q]\cdot 1 + \sum_{q\notin B} \Pr[f(X)=q]\cdot \frac{\varepsilon}{2} \\
&\qquad \le \frac{\varepsilon}{2} + \frac{\varepsilon}{2} \\
&\qquad = \varepsilon.
\end{aligned}$$

□

Now according to Lemma 2, even if Eve saved the information $f(X)$ and later obtains the private key $Y$, the distribution of $\text{EXT}(X,Y)$ is still $2^{-k}$-random with probability at least $1-2^{-k+1}$. That is, it gives an encryption scheme with everlasting security of degree $k$.

Just like previous work, we assume above that the public string $X$ is perfectly random. However, such a perfect random source may not be available in reality, and one may need to rely on sources of lower quality. Note that our proof above actually works for any source $X$ of length $n' > n$ with $H_\infty(X) \ge n$. That is, our result holds as long as Eve's storage bound is at most a fraction $\nu$ of the source's min-entropy.

### 3.1. *Reusing the Same Private Key*

Next, we show that the same key can be reused for $2^k$ times, even given all previous one-time pads. Now the idea is that for a strong extractor EXT, very likely the distribution of $Y$ would still look random even given the value of $\text{EXT}(X,Y)$, so the same $Y$ could be used again to extract randomness in the next round.

Suppose now the public random string consists of $K$ blocks $(X_1,\ldots,X_K)$ such that any block conditioned on any value of its previous blocks still has min-entropy at least $n$.[2] Consider the following scenario, with $K$ messages $M_1, M_2, \ldots, M_K$ to be transmitted. For $i \in [K]$, Alice reads the $i$th block $X_i$ of the public random string, computes the $i$th pad $Z_i = \text{EXT}(X_i, Y)$, and sends the encrypted message as $C_i = M_i \oplus Z_i$ to Bob. For $i \in [K]$, suppose Eve is also given $Z_{[i-1]} = (Z_1,\ldots,Z_{i-1})$ and uses a function $f_i$ to store $\nu n$ bits of information $Q_i = f_i(X_i, Q_{i-1}, Z_{[i-1]})$ from the public string $X_i$. Finally, Eve eavesdrops the encrypted message $C_K$ and is given $Y$, can she learn anything about $M_K$? For the security, again it suffices to show that $Z_K$ looks random even given $Z_{[K-1]}$, $Q_K$, and $Y$. We choose $\varepsilon = 2^{-3k}$, $r = \mu n$, and use a strong $(r, \varepsilon/2)$-extractor EXT. Here is the key lemma.

**Lemma 4.** *For any* $i \in \mathbb{N}$, $\|\langle Z_{[i-1]}, Q_i, Y, Z_i\rangle - \langle Z_{[i-1]}, Q_i, U^s, U^m\rangle\| \le i\varepsilon$.

[2] This is called a blockwise source, introduced by Chor and Goldreich [7].

**Proof.** We use induction on $i$. Lemma 3 handles precisely the case $i = 1$. So assume it holds for some $i \geq 1$ and consider the case $i + 1$. By triangle inequality, we have

$$\begin{aligned}\|\langle Z_{[i]}, Q_{i+1}, Y, Z_{i+1}\rangle - \langle Z_{[i]}, Q_{i+1}, U^s, U^m\rangle\| \\ \leq \|\langle Z_{[i]}, Q_{i+1}, Y, \text{EXT}(X_{i+1}, Y)\rangle - \langle Z_{[i]}, Q_{i+1}, Y', \text{EXT}(X_{i+1}, Y')\rangle\| \\ + \|\langle Z_{[i]}, Q_{i+1}, Y', \text{EXT}(X_{i+1}, Y')\rangle - \langle Z_{[i]}, Q_{i+1}, U^s, U^m\rangle\|,\end{aligned}$$

where $Y'$ is the distribution $U^s$ which is independent of $Z_{[i]}$. The first term is about how random $Y$ remains after $i$ iterations, while the second term is about how good the $(i+1)$th extraction is using a new random key $Y'$. In the first term, $Q_{i+1} = f_{i+1}(X_{i+1}, Q_i, Z_{[i]})$, and $X_{i+1}$ is independent of $Q_i$, $Z_{[i]}$, $Y$, and $Y'$. Then according to Lemma 1, the first term equals

$$\begin{aligned}\|\langle Z_{[i]}, Q_i, Y\rangle - \langle Z_{[i]}, Q_i, Y'\rangle\| &= \|\langle Z_{[i]}, Q_i, Y\rangle - \langle Z_{[i]}, Q_i, U^s\rangle\| \\ &= \|\langle Z_{[i-1]}, Q_i, Y, Z_i\rangle - \langle Z_{[i-1]}, Q_i, U^s, Z_i\rangle\| \\ &\leq i\varepsilon,\end{aligned}$$

where the inequality is from the inductive hypothesis. In the second term, $X_{i+1}$ is independent of $Z_{[i]}$, so only $Q_{i+1}$ affects the distribution of $X_{i+1}$. Then using an argument similar to the proof of Lemma 3, the second term can be bounded above by $\varepsilon$. Therefore, we have

$$\|\langle Z_{[i]}, Q_{i+1}, Y, Z_{i+1}\rangle - \langle Z_{[i]}, Q_{i+1}, U^s, U^m\rangle\| \leq (i+1)\varepsilon,$$

which proves the inductive step and thus the lemma. □

For $K \leq 2^k$, the distance is at most $2^{-2k}$. Then according to Lemma 2, even given the previous pads $Z_1, \ldots, Z_{K-1}$ and the private key $Y$, the $K$th pad $Z_K$ is still $2^{-k}$-random with probability at least $1 - 2^{-k+1}$. So we have the following theorem.

**Theorem 1.** *Any strong $(\mu n, 2^{-3k})$-extractor yields an encryption scheme with everlasting security of degree $k$, whose private key can be securely reused for $2^k$ times, against the adaptive attacks discussed above.*

Note that we only requires Eve's storage bound to be at most a $\nu$ fraction of the min-entropy in each block $X_i$. However, if we consider the $K$ blocks together, Eve's storage bound needs to be at most a $\mu/K$ fraction of the total min-entropy in the whole random string.

## 4. An On-Line Strong Extractor

Although using any strong extractor for encryption does provide a good security guarantee, not any arbitrary strong extractor is suitable in the setting we consider here. Recall that the public random string $X$ is very long so it needs to be broadcast at a very high rate. Alice and Bob do not have enough memory to store the whole string, so they would like

to be able to apply the extractor in an on-line and efficient way. One way of constructing extractors, introduced by Trevisan [19], is based on list-decodable codes. In order to have an efficient on-line strong extractor, we need a list-decodable code with an efficient on-line encoding procedure. The main work of [3] and [8] can be seen as constructing one such code, where each output bit of a codeword is a parity of a small number of input bits. Here we give another code with a better parameter.

### 4.1. *An On-Line List-Decodable Code*

We use some type of graphs called expander graphs which have their second largest eigenvalue bounded away from their largest one.[3] We also need these graphs to be explicit in the sense that given any vertex of a graph, computing its neighbors can be done efficiently.

**Lemma 5** [10], [13]. *There exists an explicit family of expander graphs $(G_n)_{n\in\mathbb{N}}$ with the following property. There is a constant $d$ and a constant $\lambda < d$ such that for every $n \in \mathbb{N}$, $G_n$ is a $d$-regular graph of $n$ vertices with the second largest eigenvalue $\lambda$.*

Expander graphs enjoys some pseudo-random properties, and we use the following one.

**Lemma 6** [2]. *Suppose $G$ is a $d$-regular graph with the second largest eigenvalue $\lambda$, and $B$ is a set containing at least an $\alpha$ fraction of $G$'s vertices. Then a $t$-step random walk on $G$ misses $B$ with probability at most $\beta_\alpha \equiv (1 - \alpha + (\lambda/d)^2)^{t/2}$.*

In fact, this probability is not far away from the probability $(1-\alpha)^t$ achieved by sampling $t$ vertices randomly and independently, but now we only need $\log n + t \log d = \log n + O(t)$ random bits for sampling, instead of $t \log n$ random bits. Objects with this property are called *oblivious samplers* (for more information, please see the survey by Goldreich [11]). Observe that smaller $\lambda/d$ gives smaller $\beta_\alpha$. This ratio could be reduced to $(\lambda/d)^i$ by considering the graph $G^i$, but then each step of a walk on $G^i$ would need $i \log d$ bits, instead of $\log d$ bits, to sample.

Let $G$ denote one such $d$-regular expander graph on $n$ vertices. For a vertex $v$ and for $w \in [d]^t$, let $W_G(v, w)$ denote the sequence of $t$ vertices visited by a $t$-step walk on $G$ starting from $v$ and then following the directions $w(1), \ldots, w(t)$. Consider the encoding $\mathrm{ECC}_t\colon \{-1, 1\}^n \to \{-1, 1\}^{\bar{n}}$, for $\bar{n} = nd^t 2^t$, defined in Fig. 1.

Note that the bit $\mathrm{ECC}_t(x)(v, w, b) = \prod_{i\in[t]} x(v_i)^{b(i)}$ is just the parity of at most $t$ input bits, those bits $x(j)$'s with $j = v_i$ and $b(i) = 1$. Suppose one had $(v, w, b)$ and derived those indices $j$ beforehand. Then when $x$ is given, the bit $\mathrm{ECC}_t(x)(v, w, b)$ can be computed in an on-line and extremely fast way.

To show that this is a good list-decodable code, we would like to bound the number of codewords within any Hamming ball of some radius. The Johnson bound in coding theory (e.g., Chapter 17 of [14]) provides one such bound for codes with some minimum distance guarantee between any two codewords. It cannot be applied directly to our code

[3] The eigenvalues of a graph's adjacency matrix are called the eigenvalues of that graph.

- **Input:** $x \in \{-1, 1\}^n$.
- **Output index:** $(v, w, b) \in [n] \times [d]^t \times \{0, 1\}^t$.
- **Algorithm:**
  — Do a $t$-step walk on $G$ to get $W_G(v, w) = (v_1, \ldots, v_t)$.
  — Output the bit indexed by $(v, w, b)$ as

$$\mathrm{ECC}_t(x)(v, w, b) \equiv \prod_{i \in [t]} x(v_i)^{b(i)}.$$

**Fig. 1.** The code $\mathrm{ECC}_t$.

$\mathrm{ECC}_t$ as some codewords are in fact close to each other.[4] However, we do have some distance guarantee between some codewords, as shown by the following lemma.

**Lemma 7.** *Suppose* $x, y \in \{-1, 1\}^n$ *and* $d_{\mathrm{H}}(x, y) \geq \alpha n$. *Then* $d_{\mathrm{H}}(\mathrm{ECC}_t(x), \mathrm{ECC}_t(y)) \geq (\frac{1}{2} - \beta_\alpha)\bar{n}$.

**Proof.** Let $B = \{v \in [n] : x(v) \neq y(v)\}$, which has at least $\alpha n$ elements. Consider any walk $(v_1, \ldots, v_t) \in [n]^t$ that hits $B$, say at $v_{i_0}$ with $x(v_{i_0}) \neq y(v_{i_0})$. For any assignment to $b$, flipping the bit $b(i_0)$ flips exactly one value of $x(v_{i_0})^{b(i_0)}$ and $y(v_{i_0})^{b(i_0)}$, and thus exactly one value of $\prod_{i \in [t]} x(v_i)^{b(i)}$ and $\prod_{i \in [t]} y(v_i)^{b(i)}$. Then for such a walk that hits $B$,

$$\Pr_{b \in \{0,1\}^t}\left[\prod_{i \in [t]} x(v_i)^{b(i)} = \prod_{i \in [t]} y(v_i)^{b(i)}\right] = \frac{1}{2}.$$

According to Lemma 6, a $t$-step random walk on $G$ misses $B$ with probability at most $\beta_\alpha$. So,

$$\begin{aligned}
&\Pr_{v,w,b}[\mathrm{ECC}_t(x)(v, w, b) = \mathrm{ECC}_t(y)(v, w, b)] \\
&\quad \leq \Pr_{v,w,b}[W_G(v, w) \text{ misses } B] \\
&\qquad + \Pr_{v,w,b}\left[\prod_{i \in [t]} x(v_i)^{b(i)} = \prod_{i \in [t]} y(v_i)^{b(i)} \mid W_G(v, w) \text{ hits } B\right] \\
&\quad \leq \beta_\alpha + \tfrac{1}{2},
\end{aligned}$$

where $W_G(v, w) = (v_1, \ldots, v_t)$ in the first inequality. Then

$$\begin{aligned}
d_{\mathrm{H}}(\mathrm{ECC}_t(x), \mathrm{ECC}_t(y)) &= (1 - \Pr_{v,w,b}[\mathrm{ECC}_t(x)(v, w, b) = \mathrm{ECC}_t(y)(v, w, b)])\bar{n} \\
&\geq (\tfrac{1}{2} - \beta_\alpha)\bar{n}.
\end{aligned}$$

□

[4] As every output bit of the code depends only on $t$ input bits, there must be an input bit that affects only $(t/n)\bar{n}$ output bits. Flipping that input bit results in two different codewords with distance at most $(t/n)\bar{n} \ll \bar{n}$, for $t \ll n$.

Now for each codeword $\text{ECC}_t(x)$, the only possible codewords with distance less than $(\frac{1}{2} - \beta_\alpha)\bar{n}$ from $\text{ECC}_t(x)$ are those $\text{ECC}_t(x')$'s with $d_{\text{H}}(x, x') \leq \alpha n$, and there are at most $2^{h(\alpha)n}$ such codewords, where $h(\alpha) = \alpha \log(1/\alpha) + (1-\alpha)\log((1/(1-\alpha))$ is the binary entropy function. Then we need the following lemma, which can be seen as a generalization to the Johnson bound, and our proof generalizes the version given in Appendix A of [5].

**Lemma 8.** *Suppose* $\text{ECC} \subseteq \{-1, 1\}^{\bar{n}}$ *is a code such that for any codeword* $c \in \text{ECC}$, *there are at most* $M$ *codewords in* ECC *within distance* $(\frac{1}{2} - \beta)\bar{n}$ *from* $c$. *Then for any* $z \in \{-1, 1\}^{\bar{n}}$ *and any* $\delta > \sqrt{\beta/2}$, *the number of codewords within distance* $(\frac{1}{2} - \delta)\bar{n}$ *from* $z$ *is at most* $M/(4\delta^2 - 2\beta)$.

**Proof.** For any two vectors $u, v \in \{-1, 1\}^{\bar{n}}$, let $u \odot v$ denote their inner product, and note that

$$u \odot v \equiv \sum_{j \in [\bar{n}]} u(j)v(j) = |\{j : u(j) = v(j)\}| - |\{j : u(j) \neq v(j)\}|.$$

Suppose $c_1 = \text{ECC}_t(x_1), \ldots, c_L = \text{ECC}_t(x_L) \in \{-1, 1\}^{\bar{n}}$ are those codewords within distance $(\frac{1}{2} - \delta)\bar{n}$ from $z$. For $i \in [L]$, let $u_i \in \{-1, 1\}^{\bar{n}}$ represent the discrepancy between $c_i$ and $z$, with $u_i(j) = c_i(j)z(j)$ for $j \in [\bar{n}]$.[5] Define

$$T \equiv \left(\sum_{i \in [L]} u_i\right) \odot \left(\sum_{i \in [L]} u_i\right).$$

On one hand,

$$\begin{aligned} T &= \sum_{j \in [\bar{n}]} \left(\sum_{i \in [L]} u_i(j)\right)^2 \\ &\geq \frac{1}{\bar{n}} \left(\sum_{j \in [\bar{n}]} \sum_{i \in [L]} u_i(j)\right)^2, \end{aligned}$$

from Cauchy–Schwartz inequality. As $\sum_{j \in [\bar{n}]} u_i(j) = c_i \odot z \geq 2\delta\bar{n}$ for any $i \in [L]$, we have

$$\begin{aligned} T &\geq \frac{1}{\bar{n}} \left(\sum_{i \in [L]} \sum_{j \in [\bar{n}]} u_i(j)\right)^2 \\ &\geq \frac{1}{\bar{n}} (2\delta\bar{n}L)^2 \\ &= 4\delta^2\bar{n}L^2. \end{aligned}$$

On the other hand, we can write

$$T = \sum_{i \in [L]} \sum_{i' \in [L]} u_i \odot u_{i'}.$$

[5] In [5] $u_i(j)$ is defined as 1 if $c_i(j) = z(j)$ and 0 otherwise. Our definition makes the proof slightly cleaner.

For $i \in [L]$, let $N(i) = \{i' \in [L] : d_{\mathrm{H}}(x_i, x_{i'}) \leq \beta n\}$. For $i' \notin N(i)$, we have $u_i \odot u_{i'} \leq 2\beta\bar{n}$. For $i' \in N(i)$, we only have $u_i \odot u_{i'} \leq \bar{n}$, but $|N(i)| \leq M$. So

$$\begin{aligned} T &= \sum_{i\in[L]} \sum_{i'\in N(i)} u_i \odot u_{i'} + \sum_{i\in[L]} \sum_{i'\notin N(i)} u_i \odot u_{i'} \\ &\leq \bar{n}ML + 2\beta\bar{n}L^2. \end{aligned}$$

Combining the two inequalities, we have $4\delta^2\bar{n}L^2 \leq T \leq \bar{n}ML + 2\beta\bar{n}L^2$, which implies $L \leq M/(4\delta^2 - 2\beta)$. □

Note that the Johnson bound is just a special case of Lemma 8 with $M = 1$. Choose $\alpha = 1 + (\lambda/d)^2 - \delta^{4/t}$ so that $\beta_\alpha \leq \delta^2$. Applying Lemma 8 with $\beta = \beta_\alpha$ and $M = 2^{h(\alpha)n}$, we immediately have the following.

**Corollary 1.** *The code* $\mathrm{ECC}_t$ *is a* $(\frac{1}{2}-\delta, 2^{h(\alpha)n}/(2\delta^2))$*-list code, for* $\alpha = 1+(\lambda/d)^2-\delta^{4/t}$.

### 4.2. *Expanding the Output*

Given any list-decodable code $\mathrm{ECC}'$: $\{-1, 1\}^n \to \{-1, 1\}^{\bar{n}}$, it is known according to [19] and [18] that one immediately gets a strong extractor $\mathrm{EXT}'$: $\{-1, 1\}^n \times [\bar{n}] \to \{-1, 1\}$ defined as

$$\mathrm{EXT}'(x, y) = \mathrm{ECC}'(x)(y).$$

That is, the extractor encodes the input string $x$ as $\mathrm{ECC}'(x)$ and projects it onto a random dimension $y$. To obtain $m$ output bits, the approach of Aumann et al. [3], [8] is to project $\mathrm{ECC}'(x)$ onto $m$ independent and random dimensions, needing $m \log \bar{n}$ random bits for sampling. That is, they use the extractor $\mathrm{EXT}''$: $\{-1, 1\}^n \times [\bar{n}]^m \to \{-1, 1\}^m$ defined as

$$\mathrm{EXT}''(x, (y_1, \ldots, y_m)) = (\mathrm{ECC}'(x)(y_1), \ldots, \mathrm{ECC}'(x)(y_m)).$$

They use a code with $\bar{n} = n^{O(k)}$, so their extractor needs a seed of length $O(mk \log n)$. Using our code $\mathrm{ECC}_t$ for some $t = O(k)$, we immediately have an extractor of the same quality but needing a seed of length only $O(m(k + \log n))$.

For large $m$, we can get a dramatic improvement by picking some pseudo-random collection of $m$ dimensions instead of $m$ random ones for projection. This is the idea behind Trevisan's extractor construction [19] and the improvement by Raz et al. [18]. The pseudo-random projection is determined by some set system defined next.

**Definition 6** [18]. A family of sets $S_1, \ldots, S_m \subseteq [s]$ is called a weak $(\ell, \rho)$-design if

- $\forall i$, $|S_i| = \ell$, and
- $\forall i$, $\sum_{j<i} 2^{|S_i \cap S_j|} \leq \rho(m-1)$.

**Lemma 9** [18]. *For every* $\ell$, $m$ *and* $\rho > 1$, *there exists a weak* $(\ell, \rho)$*-design* $S_1, \ldots, S_m \subseteq [s]$ *with* $s = \lceil \ell/\ln \rho \rceil \ell$. *Such a family can be found in time* $\mathrm{poly}(m, s)$.

For a random $y \in \{-1, 1\}^s$, such a weak $(\log \bar{n}, \rho)$-design gives a pseudo-random collection of $m$ dimensions $y(S_1), \ldots, y(S_m)$, where each $y(S_i)$ is the integer in $[\bar{n}]$ represented by the $\log \bar{n}$ bits of $y$ indexed by $S_i$.

**Lemma 10** [19], [18]. *Suppose $S_1, \ldots, S_m \subseteq [s]$ is a weak $(\log \bar{n}, \rho)$-design, and* ECC′: $\{-1,1\}^n \to \{-1,1\}^{\bar{n}}$ *is a $(\frac{1}{2} - \varepsilon/(2m), L)$-list code. Then the function* EXT: $\{-1,1\}^n \times \{-1,1\}^s \to \{-1,1\}^m$ *defined as*

$$\text{EXT}(x, y) = (\text{ECC}'(x)(y(S_1)), \ldots, \text{ECC}'(x)(y(S_m)))$$

*is a strong $(r, \varepsilon)$-extractor, for any $r \geq \rho(m-1) + \log(2L/\varepsilon)$.*

To have a strong $(\mu n, 2^{-3k})$-extractor for our encryption scheme, we use our $(\frac{1}{2} - \delta, L)$-list code $\text{ECC}_t$, with

- $\delta = 1/(2m2^{3k})$,
- $t = O(k)$ large enough and $\lambda/d = O(1)$ small enough so that $h(\alpha) < \mu$ for $\alpha = 1 + (\lambda/d)^2 - \delta^{4/t}$, and
- $L = 2^{h(\alpha)n}/(2\delta^2) = 2^{h(\alpha)n+6k+1}m^2$.

Note that $\rho(m-1) + \log(2L/\varepsilon) = \rho(m-1) + h(\alpha)n + 9k + 2 + 2\log m$. This can be made smaller than $\mu n$ for any $m \leq n^\gamma$ with constant $\gamma \in (0,1)$, by choosing a proper $\rho = n^{O(1)}$. Using such a weak $(\log \bar{n}, \rho)$-design and our $(\frac{1}{2} - \delta, L)$ list-code $\text{ECC}_t$, we have the following theorem.

**Theorem 2.** *For any constant $\gamma \in (0,1)$, there is a strong $(\mu n, 2^{-3k})$-extractor with a seed of length $O((k + \log n)^2/\log n)$ and an output of length $n^\gamma$, where each output bit is the parity of $O(k)$ input bits.*

This, together with Theorem 1, gives the encryption scheme we claim. Recall that $n$ is typically much larger than $k$. A larger $n$ provides a higher security but only costs a negligible slow-down during encryption time. If we assume or choose $n = 2^{\Omega(k)}$, then we have an encryption scheme using a private key of length only $O(\log n)$.

## Appendix A. Proof of Lemma 1

$$\begin{aligned}
&\|\langle g(A,B), B\rangle - \langle g(A,B'), B'\rangle\| \\
&\quad = \frac{1}{2}\sum_b \sum_c |\Pr[g(A,B) = c \wedge B = b] - \Pr[g(A,B') = c \wedge B' = b]| \\
&\quad = \frac{1}{2}\sum_b \sum_c |\Pr[g(A,b) = c]\Pr[B = b] - \Pr[g(A,b) = c]\Pr[B' = b]| \\
&\quad = \frac{1}{2}\sum_b \sum_c \Pr[g(A,b) = c]|\Pr[B = b] - \Pr[B' = b]| \\
&\quad = \frac{1}{2}\sum_b |\Pr[B = b] - \Pr[B' = b]| \\
&\quad = \|B - B'\|,
\end{aligned}$$

where the second equality holds because $A$ is independent of both $B$ and $B'$.

## Appendix B. Proof of Lemma 2

The expectation of $\|\langle A \mid B = b\rangle - U\|$, with $b$ sampled from the distribution $B$, is

$$\begin{aligned}
&\sum_b \Pr[B=b]\frac{1}{2}\sum_a \left|\Pr[A=a \mid B=b] - \frac{1}{|A|}\right| \\
&\quad = \frac{1}{2}\sum_{a,b}\left|\Pr[(A,B)=(a,b)] - \Pr[B=b]\frac{1}{|A|}\right| \\
&\quad \le \frac{1}{2}\sum_{a,b}\left|\Pr[(A,B)=(a,b)] - \frac{1}{|A||B|}\right| \\
&\qquad + \frac{1}{2}\sum_{a,b}\left|\Pr[B=b]\frac{1}{|A|} - \frac{1}{|A||B|}\right| \\
&\quad = \|\langle A,B\rangle - U\| + \|B-U\| \\
&\quad \le 2\varepsilon.
\end{aligned}$$

Then the lemma follows from the Markov inequality.

## References


[1] N. Alon, J. Bruck, J. Naor, M. Naor, and R. M. Roth. Construction of asymptotically good low-rate error-correcting codes through pseudo-random graphs. *IEEE Transactions on Information Theory*, vol. 38, pages 509–516, 1992.

[2] N. Alon and J. H. Spencer. *The Probabilistic Method*. Wiley Interscience, New York, 1992.

[3] Y. Aumann, Y. Z. Ding, and M. O. Rabin. Everlasting security in the bounded storage model. *IEEE Transactions on Information Theory*, vol. 48, no. 6, pages 1668–1680, 2002.

[4] Z. Bar-Yossef, O. Reingold, R. Shaltiel, and L. Trevisan. Streaming computaion of combinatorial objects. In *Proceedings of the* 17*th IEEE Annual Conference on Computational Complexity*, pages 133–142, 2002.

[5] M. Bellare, O. Goldreich, and M. Sudan. Free bits, PCPs and non-approximability—towards tight results. *SIAM Journal on Computing*, vol. 27, no. 3, pages 804–915, 1998.

[6] C. Cachin and U. Maurer. Unconditional security against memory-bounded adversaries. In *Advances in Cryptology – CRYPTO* '97, pages 292–306, Lecture Notes in Computer Science, vol. 1294, Springer-Verlag, Berlin, 1997.

[7] B. Chor and O. Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM Journal on Computing*, vol. 17, no. 2, pages 230–261, 1988.

[8] Y. Z. Ding and M. O. Rabin. Hyper-encryption and everlasting security. In *Proceedings of the* 19*th Annual Symposium on Theoretical Aspects of Computer Science*, pages 1–26, 2002.

[9] S. Dziembowski and U. Maurer. Tight security proofs for the bounded-storage model. In *Proceedings of the* 34*th Annual ACM Symposium on Theory of Computing*, pages 341–350, 2002.

[10] O. Gabber and Z. Galil. Explicit constructions of linear-sized superconcentrators. *Journal of Computer and System Sciences*, vol. 22, no. 3, pages 407–420, 1981.

[11] O. Goldreich. A Sample of Samplers: A Computational Perspective on Sampling. *Electronic Colloquium on Computational Complexity*, Technical Report TR97-020, 1997.

[12] Z. Hartman and R. Raz. On the distribution of the number of roots of polynomials and logspace explicit extractors. In *Proceedings of the* 4*th International Workshop on Randomization and Approximation Techniques in Computer Science*, pages 3–22, 2000.

[13] A. Lubotzky, R. Philips, and P. Sarnak. Explicit expanders and the Ramanujan conjecture. In *Proceedings of the* 18*th Annual ACM Symposium on Theory of Computing*, pages 240–246, 1986.

[14] F. J. MacWilliams and N. J. A. Sloan. *The Theory of Error-Correcting Codes*. North-Holland, Amsterdam, 1981.

[15] U. Maurer. Conditionally-perfect secrecy and a provably-secure randomized cipher. *Journal of Cryptology*, vol. 5, no. 1, pages 53–66, 1992.

[16] N. Nisan. Extracting randomness: how and why—a survey. In *Proceedings of the* 11*th Annual IEEE Conference on Computational Complexity*, pages 44–58, 1996.

[17] N. Nisan and D. Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, vol. 52, no. 1, pages 43–52, 1996.

[18] R. Raz, O. Reingold, and S. P. Vadhan. Extracting all the randomness and reducing the error in Trevisan's extractors. In *Proceedings of the* 31*st Annual ACM Symposium on Theory of Computing*, pages 149–158, 1999.

[19] L. Trevisan. Extractors and pseudorandom generators. *Journal of ACM*, vol. 48, no. 4, pages 860–879, 2001.

[20] S. P. Vadhan. On constructing locally computable extractors and cryptosystems in the bounded storage model. Cryptology ePrint Archive, Report 2002/162, 2002. `http://eprint.iacr.org/`.