Electronic Colloquium on Computational Complexity, Report No. 118 (2011)

Public Key Locally Decodable Codes with Short Keys*

Brett Hemenway †
Rafail Ostrovsky ‡
Martin J. Strauss§
Mary Wootters¶

September 5, 2011

Abstract

This work considers locally decodable codes in the computationally bounded channel model. The computationally bounded channel model, introduced by Lipton in 1994, views the channel as an adversary which is restricted to polynomial-time computation. Assuming the existence of IND-CPA secure public-key encryption, we present a construction of public-key locally decodable codes, with constant codeword expansion, tolerating constant error rate, with locality $\mathcal{O}(\lambda)$, and negligible probability of decoding failure, for security parameter λ. Hemenway and Ostrovsky gave a construction of locally decodable codes in the public-key model with constant codeword expansion and locality $\mathcal{O}(\lambda^2)$, but their construction had two major drawbacks. The keys in their scheme were proportional to n, the length of the message, and their schemes were based on the Φ-hiding assumption. Our keys are of length proportional to the security parameter instead of the message, and our construction relies only on the existence of IND-CPA secure encryption rather than on specific number-theoretic assumptions. Our scheme also decreases the locality from $\mathcal{O}(\lambda^2)$ to $\mathcal{O}(\lambda)$. Our construction can be modified to give a generic transformation of any private-key locally decodable code to a public-key locally decodable code based only on the existence of an IND-CPA secure public-key encryption scheme.

Keywords: public-key cryptography, locally decodable codes, bounded channel

*A preliminary version of this work appeared in the proceedings of RANDOM 2011

†E-mail: bhemen@umich.edu. Supported in part by ONR under contract N00014-11-1-0392

‡E-mail: rafail@cs.ucla.edu. This material is based upon work supported in part by NSF grants 0830803 and 09165174, US-Israel BSF grant 2008411, grants from OKAWA Foundation, IBM, Lockheed-Martin Corporation and the Defense Advanced Research Projects Agency through the U.S. Office of Naval Research under Contract N00014-11-1-0392. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

§E-mail: martinjs@umich.edu. Supported in part by NSF grant CCF 0743372 and DARPA/ONR grant N66001-08-1-2065

¶E-mail: wootters@umich.edu

ISSN 1433-8092

1 Introduction

Error-correcting codes were designed to facilitate message transmission through noisy channels. An error-correcting code consists of two algorithms, an encoding algorithm which takes a message and adds redundancy transforming it into a (longer) codeword. A decoding algorithm takes a (corrupted) codeword and recovers the original message. Although error-correcting codes were designed for data transmission, they have seen widespread use in storage applications. In storage environments, *random access* to the data is often of importance. A code that can recover a single bit of the underlying message by reading a small number of bits of a corrupted codeword is called *locally decodable*. Locally decodable codes were introduced in the context of Probabilistically-Checkable Proofs (PCPs) [BFLS91, Sud92, PS94], and were formalized explicitly in the work of Katz and Trevisan [KT00]. In current applications, where random access is needed, the underlying data is broken into small blocks and each block is encoded separately using a standard error-correcting code. This technique allows for local recovery, since to recover a single bit of the message, the decoder needs to only read a single block. However, a small number of errors concentrated on a single block may destroy part of the message. Locally decodable codes can be seen as a way to achieve the best of both worlds: the robustness of encoding the database as a single codeword, while still providing random-access into the underlying message. These benefits come at a price, and Despite significant research (see [Tre04, Yek10] for surveys) locally decodable codes have much larger codeword expansion than their classical counterparts. The most efficient 3-query LDCs are given by Efremenko [Efr09], and have codeword expansion of $\exp(\exp(\mathcal{O}(\sqrt{\log n \log\log n})))$ for messages of length n. While these codes have found many applications towards Probabilistically-Checkable Proofs and Private Information Retrieval, their expansion rate is far too large for data storage applications. Recent work by Kopparty, Saraf and Yekhanin [KSY11] gives constant rate locally decodable codes with locality $\mathcal{O}(n^\epsilon)$. These codes provide a drastic reduction in codeword expansion at the price of fairly high locality.

There are a number of models for the introduction of errors. Shannon's original work [Sha48], considered errors that were introduced by a binary symmetric channel, where every by of a codeword was independently "flipped" with some constant probability. This model is relatively weak; a significantly stronger model is Hamming's adversarial model. In the adversarial model, the channel is viewed as an adversary who is attempting to corrupt a codeword. The channel's only limitation is on the number of symbols it is allowed to corrupt. Shannon's random errors and Hamming's worst-case errors provide two extreme models, and much work has gone into designing codes that are robust against some intermediate forms of error.

We will focus on the computationally-bounded channel model proposed by Lipton [Lip94, GLD04]. In this model, like in Hamming's model, we view the channel as an adversary who is attempting to cause a decoding error. As in Hamming's model the channel is restricted in the number of symbols (or bits) it can corrupt, but we further restrict the channel to feasible (polynomial-time) computations. This computationally-bounded channel model has been studied in the context of classical error-correction [Lip94, GLD04, MPSW05], and locally decodable codes [OPS07, HO08].

In this work, we present a construction of locally decodable codes in the computationally-

bounded channel model with constant codeword expansion and locality $\mathcal{O}(\lambda)$ where λ is the security parameter. In addition to improved locality, our results offer significant improvements over previous constructions constructions of locally decodable codes in the computationally-bounded channel model. Our codeword expansion matches that of [HO08], but we address the two main drawbacks of that construction. Our keys are much shorter ($\mathcal{O}(\lambda)$ instead of $\mathcal{O}(n)$), and our construction requires only the existence of an IND-CPA secure cryptosystem, while their result relies on the relatively strong Φ-hiding assumption [CMS99].

1.1 Previous Work

The computationally bounded channel model was introduced by Lipton [Lip94, GLD04], where he showed how a shared key can reduce worst-case (adversarial) noise to random noise. Lipton's construction worked as follows. The sender and receiver share a permutation $\sigma \in S_n$, and a blinding factor $r \in \{0,1\}^n$. If $\mathbf{ECC}$ is an error-correcting code with codewords of length n, robust against random noise, then $m \mapsto \sigma(\mathbf{ECC}(m)) \oplus r$ is an encoding robust against adversarial noise. If the channel is not polynomially-bounded the sender and receiver must share $n \log n + n$ bits to communicate an n-bit codeword. If, however, the channel is polynomially-bounded, and one-way functions exist, then the sender and receiver can share a (short) seed for a pseudo-random generator rather than the large random objects σ and r.

One drawback of Lipton's construction is that it requires the sender and receiver to share a secret key. In [MPSW05], Micali, Peikert, Sudan and Wilson considered public-key error correcting codes against a bounded channel. They observed that if $\mathsf{Sign}(sk, \cdot)$ is an existentially unforgeable signature scheme, and $\mathbf{ECC}$ is a *list-decodable* error correcting code, then $\mathbf{ECC}(m, \mathsf{Sign}(sk, m))$ can tolerate errors up to the list-decoding bound of $\mathbf{ECC}$ against a computationally bounded channel. The receiver needs only to list decode the corrupted codeword and choose the item in the list with a valid signature. Since the channel is computationally bounded it cannot produce valid signatures, so with all but negligible probability there will be only one message in the list with a valid signature. This technique allowed them to create codes that could decode beyond the Shannon bound.

A new twist on the code-scrambling technique was employed by Guruswami and Smith [GS10] to construct optimal rate error correcting codes against a bounded channel in the setting where the sender and receiver do not share a key (and there is no public-key infrastructure). In the Guruswami and Smith construction, the sender chooses a random permutation and blinding factor, but then embeds this "control information" into the codeword itself and sends it along with the message. The difficulty lies in designing the code so that the receiver can parse the codeword and extract the control information which then allows the receiver to recover the intended message (called the "payload information"). Guruswami and Smith's codes work in a more limited channel model, requiring the channel to be oblivious (independent of the actual codeword being sent) or restricted to using logarithmic space, and their codes are not locally decodable. This work considers a stronger channel model (polynomial time) and considers codes that are locally decodable. Unlike those of Guruswami and Smith, however, our codes require setup assumptions (a public-key infrastructure) and only achieve constant (not optimal) rate.

Locally decodable codes were first studied in the computationally bounded channel model by Ostrovsky, Pandey and Sahai [OPS07]. In their work, they showed how to adapt Lipton's code-scrambling to achieve locally decodable codes when the sender and receiver share a secret key. Their constructions achieved constant ciphertext expansion and locality $\omega(\log^2 \lambda)$.

In [HO08], Hemenway and Ostrovsky considered locally decodable codes in the public-key setting. They used Private Information Retrieval (PIR) to implement a hidden permutation in the public-key model. Their construction achieves constant ciphertext expansion, and locality $\mathcal{O}(\lambda^2)$. Their construction suffered from two major drawbacks, first the key-size was $\mathcal{O}(n)$ since it consisted of PIR queries implementing a hidden permutation, and second the only one of their constructions to achieve constant ciphertext expansion was based on the Φ-hiding assumption [CMS99]. Prior to this work, however, these were the only locally decodable codes in the public-key model.

The work of Bhattacharyya and Chakraborty [BC11] considers locally decodable codes in the bounded channel model, but their work concerns negative results. They show that public-key locally decodable codes with *constant* locality and linear decoding algorithm must be smooth, and hence the restriction on the channel does not make the constructions easier. The codes constructed in this paper have a non-linear decoding algorithm as well as super-constant locality, so the negative results of [BC11] do not apply.

1.2 Our Contributions

We address the problem of constructing locally decodable codes in the public key computationally bounded channel model. Prior to this work, the best known constructions of locally decodable codes in the computationally bounded channel model were due to Hemenway and Ostrovsky [HO08]. While both their construction and ours yield locally decodable codes in the computationally bounded channel model with constant codeword expansion, our construction has a number of significant advantages over the previous constructions.

- **Size of keys:**
  For security parameter λ, and messages of length n, our construction has keys that are size $\mathcal{O}(\lambda)$, while [HO08] has keys that are of size $\mathcal{O}(n)$, indeed, this is a primary drawback of their scheme.

- **Locality:**
  For security parameter λ, and messages of length n, Our construction has locality $\mathcal{O}(\lambda)$, improving the locality $\mathcal{O}(\lambda^2)$ in [HO08]. To recover a single bit (or $\mathcal{O}(\lambda)$) bits of a message from a corrupted codeword requires reading $\mathcal{O}(\lambda)$ bits from the codeword. This improves the locality of the construction in [HO08] which had locality $\mathcal{O}(\lambda^2)$ for the same parameters.

- **Generality:**
  The scheme of [HO08] only achieves constant codeword expansion under the Φ-hiding assumption, while our schemes require only the existence of IND-CPA secure encryption. Like [OPS07, HO08], our codes have constant ciphertext expansion and fail to decode with negligable probability.

- **Re-usability:**
  In previous schemes, relying on a hidden permutation [Lip94, GLD04, OPS07, HO08], the permutation is fixed by the key, and thus an adversary who monitors the bits read by the decoder can efficiently corrupt future codewords.[1] In the private key setting [Lip94, GLD04, OPS07] this can be remedied by forcing the sender and receiver to keep state. Public-key schemes which rely on a hidden permutation cannot be modified in the same way to permit re-usability. Indeed, even in the case of codes without local decodability creating optimal rate codes in the bounded channel model that do not require sender and receiver to keep state was indicated as a difficult problem in [MPSW05].[2]

- **Codeword Size:**
  Our codes have constant ciphertext expansion, matching the results of [HO08] in the public-key setting and [OPS07] in the private-key setting.

- **Probability of Incorrect Decoding:**
  In the computationally-bounded channel model it is standard to consider codes that only fail to decode with negligible probability. This is the case for our construction, as well as the constructions of [OPS07, HO08]. In the unbounded error model it is traditional to consider codes with *constant* failure probability. We note, however, that simply repeating the decoding and taking a majority vote can be used to decrease the failure probability at cost of increasing the locality.

These claims require that the message length n be greater than λ^2, (where λ is the security parameter). This is a minor restriction, however, since the Locally Decodable Codes are aimed at settings where the messages are large databases.

As in [Lip94, GLD04, OPS07, HO08] our construction can be viewed as a permutation followed by a blinding. In these types of constructions, the difficulty is how the sender and receiver can agree on the permutation and the blinding factor. The blinding can easily be achieved by standard PKE, so the primary hurdle is how the sender and receiver can agree on the permutation. In [OPS07] the sender and receiver were assumed to have agreed on the permutation (or a seed for a pseudo-random permutation) prior to message transmission (this is the secret-key model). In [HO08], the receiver was able to hide the permutation in his public-key by publishing PIR queries for the permutation. This has the drawback that the public-key size must be linear in the length of the message. In both [OPS07, HO08], the permutation is fixed and remains the same for all messages. In this work we take a different approach, similar to that of [GS10]. The sender generates a fresh (pseudo) random permutation for each message and encodes the permutation into the message itself. Codewords consist of two portions, the control portion (which specifies the permutation) and the payload portion (which encodes the actual message). The difficulty is in showing that the adversary cannot corrupt either the control portion or the payload portion, even if the decoder only reads a small portion of the control information.

[1] This notion of re-usability is different than [OPS07], where they call a code re-usable if it remains secure against an adversary who sees multiple codewords, but who cannot see the read pattern of the decoder.

[2] Our solution does not solve the problem posed in [MPSW05], however, because while our codes transmit data at a constant rate, they do not achieve the Shannon capacity of the channel.

1.3 Notation

If $f : X \to Y$ is a function, for any $Z \subset X$, we let $f(Z) = \{f(x) : x \in Z\}$. If A is a PPT machine, then we use $a \xleftarrow{\$} A$ to denote running the machine A and obtaining an output, where a is distributed according to the internal randomness of A. If R is a set, and no distribution is specified, we use $r \xleftarrow{\$} R$ to denote sampling from the uniform distribution on R. We say that a function ν is negligible if $\nu = o(n^{-c})$ for every constant c. For a string x, we use $|x|$ to denote the length (in bits) of x. For two strings $x, y \in \{0,1\}^n$ we use $x \oplus y$ to denote coordinate-wise exclusive-or.

2 Locally Decodable Codes

In this section we define the codes and channel model we consider. The notion of local-decodability arose in the construction of probabilistically-checkable proofs (PCPs), see for example [BFLS91, Sud92, PS94]. The first formal definition of locally decodable codes was put forward by Katz and Trevisan in [KT00]. Since then the study of locally decodable codes has grown significantly. Good surveys of the study of locally decodable codes are available [Tre04, Yek10].

Definition 1 (Adversarial Channels)**.** An adversarial channel of error rate δ is a randomized map $A : \{0,1\}^n \to \{0,1\}^n$ such that for all w, $\text{dist}(w, A(w)) < \delta n$. We say that the channel is *computationally bounded* if A can be computed in time polynomial in n.

Definition 2 (Locally Decodable Codes)**.** A code $\mathbf{ECC} = (\mathbf{ECCEnc}, \mathbf{ECCDec})$ is a called a $[q, \delta, \epsilon]$ locally decodable code with rate r if for all adversarial channels A of error rate δ we have

- For all x, and all $i \in [k]$ it holds that $\Pr[\mathbf{ECCDec}(A(x), i) = x_i] \geq 1 - \epsilon$.
- $\mathbf{ECCDec}$ makes at most q queries to $A(x)$.
- The ratio $|x|/|\mathbf{ECCEnc}(x)| = r$.

Where x_i denotes the ith bit of x.

Simply letting A be computationally bounded in Definition 2 is not sufficient since it does not address A's ability to see the public-key or adapt to previous read patterns.

Definition 3 (Public-Key Locally Decodable Codes)**.**
A code $\mathbf{PKLDC} = (\mathbf{PKLDCGen}, \mathbf{PKLDCEnc}, \mathbf{PKLDCDec})$ is a called a $[q, \delta, \epsilon]$ public-key locally decodable code with rate r if all polynomial time adversarial channels A of error rate δ have probability at most ϵ of winning the following game. The game consists of three consecutive phases.

1. **Key Generation Phase:**
   The challenger generates $(pk, sk) \xleftarrow{\$} \mathbf{PKLDCGen}(1^\lambda)$, and gives pk to the adversary.

2. **Query Phase:**
   The adversary can adaptively ask for encodings of messages x, and receives $c = \mathbf{PKLDCEnc}(pk, x)$. For any $i \in [n]$, the adversary can then ask for the decoding of the ith bit of x from c, and learn the q indices in c that were queried by $\mathbf{PKLDCDec}(sk, c, i)$.

3. **Challenge Phase:**
   The adversary chooses a challenge message x, and receives $c = \mathbf{PKLDCEnc}(pk, x)$, the adversary outputs $\tilde{c}$. The adversary wins if $|\tilde{c}| = |c|$, $\text{dist}(\tilde{c}, c) \leq \delta|c|$, and there exists an $i \in [n]$ such that $\mathbf{PKLDCDec}(sk, \tilde{c}, i) \neq x_i$.

We also require that

- $\mathbf{PKLDCDec}(sk, c)$ makes at most q queries to the codeword c.
- The ratio $|x|/|\mathbf{PKLDCEnc}(pk, x)| = r$.

We will consider locally decodable codes in the computationally bounded channel model, and we will focus on the case where the error rate δ is constant, the transmission rate r is constant. If we specify that the probability ϵ of decoding error is a negligible function of the security parameter, then with these constraints our goal is to minimize the locality q.

Remark: In the query phase, we allowed the adversary to see indices read by the challenger when decoding a codeword created by the challenger itself. We could allow the adversary to see the indices read by decoding algorithm on any string c. Proving security in this more general setting could be achieved using the framework below by switching the IND-CPA encryption scheme in our construction for an IND-CCA one.

3 Construction

Let $\mathcal{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be a semantically secure public-key encryption, with plaintexts of length 2λ, and ciphertexts of length $2d\lambda$. Let $\mathbf{ECC}_1 = (\mathbf{ECCEnc}_1, \mathbf{ECCDec}_1)$ be an error correcting code with $2d\lambda$ bit messages and $2dd_1\lambda$ bit codewords. Let $\mathbf{ECC}_2 = (\mathbf{ECCEnc}_2, \mathbf{ECCDec}_2)$ be an error correcting code with t bit messages and $d_2 t$ bit codewords, and let PRG be a pseudo-random generator taking values in the symmetric group on $d_2 n$ symbols. Thus $\mathsf{PRG}(\cdot) : \{0,1\}^\lambda \to S_{d_2 n}$. Let $\widetilde{\mathsf{PRG}}$ be a pseudo-random generator from $\{0,1\}^\lambda \to \{0,1\}^{d_2 n}$.

- **Key Generation:**
  The algorithm $\mathbf{PKLDCGen}(1^\lambda)$ samples $(pk, sk) \xleftarrow{\$} \mathsf{Gen}$. The public key will be pk along with the two function descriptions $\mathsf{PRG}, \widetilde{\mathsf{PRG}}$, while the secret key will be sk.

- **Encoding:**
  To encode a message $m = m_1 \cdots m_n$, the algorithm $\mathbf{PKLDCEnc}$ breaks m into blocks of size t and set $c_i = \mathbf{ECCEnc}_2(m_i)$ for $i = 1, \ldots, n/t$. Set $C = c_1 \cdots c_{n/t}$, so $|C| = d_2 n$. Sample $x_1 \xleftarrow{\$} \{0,1\}^\lambda$. $x_2 \xleftarrow{\$} \{0,1\}^\lambda$, and let $\sigma = \mathsf{PRG}(x_1)$, and

$R = \widetilde{\mathsf{PRG}}(x_2)$. Generate $r \xleftarrow{\$} \mathsf{coins}(\mathsf{Enc})$. The codeword will be

$$(\underbrace{\mathbf{ECCEnc}_1(\mathsf{Enc}((x_1,x_2),r)),\ldots,\mathbf{ECCEnc}_1(\mathsf{Enc}((x_1,x_2),r))}_{\ell \text{ copies}}, R \oplus \sigma(C)).$$

So a codeword consists of ℓ copies of the "control information" $\mathbf{ECCEnc}_1(\mathsf{Enc}((x_1,x_2),r))$, followed by the "payload information" $R \oplus \sigma(C)$.

- **Decoding:**
  The algorithm **PKLDCDec** takes as input a codeword $(c_1,\ldots,c_\ell,P)$, and a desired block $i^* \in \{1,\ldots,n/t\}$. First, the decoder must recover the control information. For j from 1 to $2dd_1\lambda$, **PKLDCDec** chooses a block $i_j \in [\ell]$, and reads the jth bit from the i_jth control block. Concatenating these bits, the decoder has (a corrupted version) of $c = \mathbf{ECCEnc}_1(\mathsf{Enc}((x_1,x_2),r))$. The decoder decodes with $\mathbf{ECCDec}_1$, and then decrypts using Dec to recover (x_1,x_2). The control information (x_1,x_2) will be recovered correctly if no more than a δ_1 fraction of the bits $2dd_1\lambda$ bits read by the decoder were corrupted. Second, once the decoder has the control information. The decoder then recovers $\sigma = \mathsf{PRG}(x_1)$, and $R = \widetilde{\mathsf{PRG}(x_2)}$. The block i^* consists of the bits $i^*t,\ldots,(i^*+1)t-1$ of the message m, so the decoder reads the bits $P_{\sigma(i^*d_2t)},\ldots,P_{\sigma(i^*+1)d_2t-1}$ from the received codeword. The decoder then removes the blinding factor

  $$C = P_{\sigma(i^*d_2t)} \oplus R_{\sigma(i^*d_2t)} \cdots P_{\sigma((i^*+1)d_2t-1)} \oplus R_{\sigma((i^*+1)d_2t-1)}$$

  At this point C is a codeword from $\mathbf{ECC}_2$, so the decoder simply outputs $\mathbf{ECCDec}_2(C)$. The locality is $2dd_1\lambda + d_2t$.

$\mathcal{PKE}$	IND-CPA encryption with 2λ bit messages and $2d\lambda$ bit ciphertexts
$\mathbf{ECC}_1$	Error Correcting code with $2d\lambda$ bit messages and $2dd_1\lambda$ bit codewords
$\mathbf{ECC}_2$	Error Correcting code with t bit messages and d_2t bit codewords
PRG	A pseudo random generator, outputting a permutation in S_{d_2n}
$\widetilde{\mathsf{PRG}}$	A pseudo random generator, outputting a string in $\{0,1\}^{d_2n}$
ℓ	The number of times the control information is repeated
t	The blocksize of the payload information

Figure 1: Constituents of the code

Remarks: The above scheme admits many modifications. In particular, there are a number of simple tradeoffs that can be made to increase the correctness of the scheme, while decreasing the locality. The simplest way to increase the probability that an uncorrupted copy of (x_1,x_2) is chosen, is to increase the number of code blocks read by the decoder. A slightly different approach to increase the probability that an uncorrupted copy of (x_1,x_2)

is chosen, is to have the sender sign each encryption $\mathsf{Enc}(x_1, x_2, r)$, the receiver could then read back values until one with a valid signature is found. This would require the receiver to have the sender's verification key, and would result in an scheme with better *average* locality. Tradeoffs of this sort between locality (or codeword expansion) and correctness are commonplace in coding theory, and we make no attempt to list them all here.

- **Codeword Length:** A codeword is of the form

$$(\underbrace{\mathbf{ECCEnc}_1(\mathsf{Enc}((x_1,x_2),r)),\ldots,\mathbf{ECCEnc}_1(\mathsf{Enc}((x_1,x_2),r))}_{2\ell d d_1\lambda \text{ bits}},\underbrace{R\oplus\sigma(C)}_{d_2 n \text{ bits}}).$$

  Thus the total codeword length is $2\ell d d_1\lambda + d_2 n$, making the codeword expansion $\frac{2\ell d d_1\lambda + d_2 n}{n}$.

- **Locality:** The locality is $2dd_1\lambda + d_2 t$. If we take $t = \mathcal{O}(\lambda)$, then we will successfully recover with all but negligible probability (negligible ϵ), and the locality will be $\mathcal{O}(\lambda)$.

Theorem 1. The scheme $\mathbf{PKLDC} = (\mathbf{PKLDCGen}, \mathbf{PKLDCEnc}, \mathbf{PKLDCDec})$ is a public-key locally decodable code with locality $q = 2dd_1\lambda + d_2 t$, and error rate δ, with failure probability

$$\epsilon = \left(e^{\frac{\delta_1}{\alpha_1}-1} \Big/ \left(\frac{\delta_1}{\alpha_1}\right)^{\frac{\delta_1}{\alpha_1}} \right)^{2\alpha_1 dd_1\lambda} + ne^{-2\frac{(\delta_2-\alpha_2)^2 d_2^2 t^2 - 1}{d_2 t+1}} + \nu(\lambda)$$

for some negligible function $\nu(\cdot)$. Where α_1, α_2 are any numbers with $0 \le \alpha_1, \alpha_2 \le 1$, satisfying

$$2\alpha_1 dd_1\lambda\ell + \alpha_2 d_2 n \le \delta|C| = 2\delta\ell d d_1\lambda + \delta d_2 n,$$

and δ_i is the error rate tolerated by $\mathbf{ECC}_i$ for $i \in \{1,2\}$. In particular, this means that for all PPT algorithms A and for all i

$$\Pr\left[\begin{array}{l} \mathbf{PKLDCDec}(sk,\tilde{C},i) \neq x_i : \\ \quad (pk,sk) \xleftarrow{\$} \mathbf{PKLDCGen}(1^\lambda) \\ \quad C \xleftarrow{\$} \mathbf{PKLDCEnc}(pk,x), \tilde{C} \xleftarrow{\$} A(C,pk) \end{array}\right] < \epsilon$$

whenever C and $\tilde{C}$ have the same length, and differ in at most $\delta|C|$ bits.

Proof. Since the codewords are naturally divided into two types of information, *control information*, and *payload information*, we distinguish between errors in each type. Let $\mathfrak{e}^c$ be the event that the adversary succeeds in corrupting the control information read by the decoder, and let $\mathfrak{e}_i^p$ be the event that the adversary succeeds in corrupting payload block i. Given a corrupted codeword $\tilde{C} = (\tilde{c}_1,\ldots,\tilde{c}_\ell,\tilde{P})$, $\mathfrak{e}^c$ is the event that more than a δ_1 fraction of the $2dd_1\lambda$ control bits are corrupted, so the event $\mathfrak{e}^c$ corresponds to the event that the adversary succeeds in making the decoder recover erroneous control information. Similarly, $\mathfrak{e}_i^p$ is the event that more than a δ_2 fraction of the bits of the payload block $P_{\sigma(id_2t)}\cdots P_{\sigma((i+1)d_2t-1)}$ are corrupted. Recall that δ_i is the error tolerance of $\mathbf{ECC}_i$, in

particular, $\mathbf{ECC}_i$ successfully decodes from a δ_i fraction of corrupted bits. It is easy to see that the probability of incorrect decoding is bounded above by $\Pr[\mathfrak{e}^c] + \sum_i \Pr[\mathfrak{e}_i^p])$.

We proceed via a series of games, and we argue that $\Pr[\mathfrak{e}^c]$ and $\Pr[\mathfrak{e}_i^p]$ are essentially the same in each game.

game_0 This is the actual corruption game.

game_1 This game is identical to game_0 except that we imagine the challenger playing the role of both sender and receiver.

game_2 This game is identical to game_1 except that when decoding, the challenger selects indices $i_1, \ldots, i_{2dd_1\lambda} \in [\ell]$, and if more than δ_1 fraction of the bits specified by them are incorrect, the challenger outputs $\perp$, otherwise the challenger continues to read the appropriate payload blocks. Notice that the challenger no longer needs sk since it does not decrypt the control block, but merely checks if it has been corrupted. This can be done by a challenger who simply stores the uncorrupted codeword and compares it to the corrupted word outputted by the channel.

game_3 This game is identical to game_2 except that when encoding the challenger does not encrypt (x_1, x_2), but $(0, 0)$, so in this game codewords look like

$$(\mathbf{ECCEnc}_1(\mathsf{Enc}((0,0), r_1)), \ldots, \mathbf{ECCEnc}_1(\mathsf{Enc}((0,0), r_\ell)), R \oplus \sigma(C)).$$

Notice that the challenger can still recognize the events $\mathfrak{e}^c$ and $\mathfrak{e}_i^p$, by storing the uncorrupted codeword and comparing it to the corrupted codeword without the need for sk.

game_4 This is identical to game_3 except that in this game the challenger generates σ uniformly from $S_{d_2 n}$ and R uniformly from $\{0,1\}^{d_2 n}$.

From an adversary's perspective, game_0 and game_1 are identical. In game_2, the probability of decryption failure may increase, but $\Pr[\mathfrak{e}^c]$ and $\Pr[\mathfrak{e}_i^p]$ remain unchanged. By the semantic security of $\mathcal{PKE}$, $\Pr[\mathfrak{e}^c]$ and $\Pr[\mathfrak{e}_i^p]$ can only change by a negligible amount between game_2 and game_3. By the security of PRG, $\Pr[\mathfrak{e}^c]$ and $\Pr[\mathfrak{e}_i^p]$ can only change by a negligible amount between game_3 and game_4.

Thus to prove the claim it remains only to bound $\Pr[\mathfrak{e}^c]$ and $\Pr[\mathfrak{e}_i^p]$ in game_4. To bound $\Pr[\mathfrak{e}^c]$ and $\Pr \mathfrak{e}_i^p]$ in this game, suppose A introduces α_1 fraction of errors into the control information and α_2 error into the payload information. Since the adversary introduces at most a δ fraction of errors into the entire codeword, we have

$$2\alpha_1 dd_1\lambda\ell + \alpha_2 d_2 n \leq \delta|C| = 2\delta\ell dd_1\lambda + \delta d_2 n$$

Recall that the control information $\mathbf{ECCEnc}_1(\mathsf{Enc}((x_1, x_2), r))$ is $2dd_1\lambda$ bits long, and there are ℓ copies of it in the codeword. Let Z_j denote the event that the jth control bit read by the decoder is corrupted, where the probability ranges over the decoder's choice over which of the ℓ copies the bit is read from. Then each Z_j is an independent Bernoulli random

variable, and $\sum_{i=1}^{2dd_1\lambda} \mathbb{E}(Z_i) = 2\alpha_1 dd_1\lambda$. A Chernoff bound yields

$$\Pr[\mathfrak{e}^c] = \Pr\left[\sum_{j=1}^{2dd_1\lambda} Z_j > 2\delta_1 dd_1\lambda\right] < \left(e^{\frac{\delta_1}{\alpha_1}-1} \Big/ \left(\frac{\delta_1}{\alpha_1}\right)^{\frac{\delta_1}{\alpha_1}}\right)^{2\alpha_1 dd_1\lambda}$$

We observe that this will clearly be negligible in λ, whenever $\delta_1 > \alpha_1$, i.e. the error tolerance of $\mathbf{ECC}_1$ is greater than the proportion of the control information that is corrupted. By choosing ℓ to be large enough and δ to be small enough, we can always ensure that this is the case.

To analyze the probability that the adversary successfully corrupts a payload block, we observe that since σ and R are uniform, the adversary's corruptions are distributed uniformly among the $d_2 n$ payload bits. The number of errors in a given payload block is distributed according the hypergeometric distribution with parameters $(\alpha_2 d_2 n, d_2 n, d_2 t)$.

Theorem 1 from [HS05] gives

$$\Pr[\mathfrak{e}_i^p] = \Pr[\#\text{errors in block i} > \delta_2 d_2 t] < e^{-2\frac{(\delta_2-\alpha_2)^2 d_2^2 t^2 - 1}{d_2 t + 1}}.$$

It is easy to see that if $\delta_2 > \alpha_2$, then this drops exponentially quickly in t. □

Corollary 1. If there exists IND-CPA secure encryption with constant ciphertext expansion then for messages of length $n \geq \lambda^2/2$ there exists Public-Key Locally Decodable Codes of constant rate tolerating a constant fraction of errors with locality $q = \mathcal{O}(\lambda^2)$, and $\epsilon = \nu(\lambda)$ for some negligible function ν.

Proof. Taking $\ell = n/\lambda$ and $t = \lambda$, in the above construction, we have codeword expansion of $2dd_1 + d_2$, which is a constant depending only on the expansion of the encryption (d) and the expansion of the two error correcting codes (d_1, d_2). The code has locality $(dd_1 + d_2)\lambda$.

The code recovers with probability

$$\epsilon = \left(e^{\frac{\delta_1}{\alpha_1}-1} \Big/ \left(\frac{\delta_1}{\alpha_1}\right)^{\frac{\delta_1}{\alpha_1}}\right)^{2\alpha_1 dd_1\lambda} + ne^{-2\frac{(\delta_2-\alpha_2)^2 d_2^2 t^2 - 1}{d_2 t + 1}} + \nu(\lambda)$$

when $\delta_i > \alpha_i$ for $i \in \{1, 2\}$. With $t = \lambda$, ϵ will be negligible in λ. So we just need to assure that $\delta_i > \alpha_i$. Recalling that α_1, α_2 were the fraction of errors in the control and payload blocks respectively, we have the following relationship

$$2\alpha_1 dd_1\lambda\ell + \alpha_2 d_2 n \leq \delta|C| = 2\delta(\ell dd_1\lambda + d_2 n) = \delta(2dd_1 + d_2)n$$

With $\ell = n\lambda$, this yields the trivial bounds:

$$\alpha_1 \leq \delta\frac{2dd_1 + d_2}{2dd_1}, \qquad \alpha_2 \leq \delta\frac{2dd_1 + d_2}{d_2}.$$

So it will be enough to choose δ such that

$$\delta\frac{2dd_1 + d_2}{2dd_1} \leq \delta_1, \qquad \delta\frac{2dd_1 + d_2}{d_2} \leq \delta_2.$$

This yields

$$\delta \leq \min\left(\frac{2dd_1\delta_1}{2dd_1+d_2}, \frac{d_2\delta_2}{2dd_1+d_2}\right).$$

Since δ_1, δ_2 are the constant error rates tolerated by the underlying error correcting codes, and d_1, d_2 are the constant expansion rates of the underlying error correcting codes, the error rate tolerated by **PKLDC** will also be constant.

To give a sense of the efficiency of this scheme, we can plug in concrete numbers. If $\mathcal{PKE}$ is a cryptosystem with ciphertexts twice as long as messages (i.e. $d = 2$), and the two error correcting codes can tolerate a 1/16 fraction of errors ($\delta_1 = \delta_2 = 1/16$) with rate 1/4, ($d_1 = d_2 = 4$), then **PKLDC** can tolerate an error rate of $\delta = \frac{1}{80}$, with ciphertext expansion of 20. □

A similar construction works to convert any Secret Key Locally Decodable Code [OPS07] to a PKLDC using only a standard IND-CPA secure cryptosystem.

4 Construction Based on Secret-Key Locally Decodable Codes

In this section, we show how to transform any Secret-Key Locally Decodable code for messages of length $n > \lambda^2$, into a public-key locally decodable code with similar parameters.

Let $\mathcal{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be a semantically-secure public-key cryptosystem encrypting messages of length λ, and ciphertexts of length $d\lambda$. Let $\mathbf{ECC}_1 = (\mathbf{ECCEnc}_1, \mathbf{ECCDec}_1)$ be an error correcting code with $d\lambda$ bit messages and $dd_1\lambda$ bit codewords. Let $\mathbf{SKLDC} = (\mathbf{SKLDCGen}, \mathbf{SKLDCEnc}, \mathbf{SKLDCDec})$ is a secret-key locally decodable code with keys of length λ, encoding messages of length n, and locality q, codeword expansion d_2 tolerating an error rate of δ_2.

We construct a public-key locally decodable code as follows:

- **Key Generation:**
  The algorithm $\mathbf{PKLDCGen}(1^\lambda)$ samples $(pk, sk) \xleftarrow{\$} \mathsf{Gen}$. The public key will be pk, and the secret key will be sk.

- **Encoding:**
  To encode a message m, generate $sk' \xleftarrow{\$} \mathbf{SKLDCGen}(1^\lambda)$, For $i \in [n]\ell$,

  Generate $r \xleftarrow{\$} \mathsf{coins}(\mathsf{Enc})$, and the codeword will be

  $$(\underbrace{\mathbf{ECCEnc}_1(\mathsf{Enc}(sk', r)), \ldots, \mathbf{ECCEnc}_1(\mathsf{Enc}(sk', r))}_{\ell \text{ times}}, \mathbf{SKLDCEnc}(sk', m)).$$

- **Decoding:**
  The algorithm **PKLDCDec** takes as input a codeword

  $$(c_1, \ldots, c_\ell, P),$$

  and a desired block $i^* \in \{1, \ldots, n/t\}$.

First, the decoder must recover the control information. For j from 1 to $dd_1\lambda$, **PKLDCDec** chooses a block $i_j \in [\ell]$, and reads the jth bit from the i_jth control block. Concatenating these bits, the decoder has (a corrupted version) of $c = \mathbf{ECCEnc}_1(\mathsf{Enc}(sk', r))$. The decoder decodes with $\mathbf{ECCDec}_1$, and then decrypts using Dec to recover (sk'). The control information (sk') will be recovered correctly if no more than a δ_1 fraction of the bits $dd_1\lambda$ bits read by the decoder were corrupted.

Second, using the recovered key sk', the decoder runs the local decoding procedure $\mathbf{SKLDC}(sk', P, i^*)$. The locality is $dd_1\lambda + q$ where q is the locality of **SKLDC**.

Theorem 2. The construction above is a Public-Key Locally Decodable code with locality $dd_1\lambda + q$, and codewords of size $dd_1\lambda\ell + d_2 n$ tolerating error rate

$$\delta \leq \min\left(\frac{\delta_1 dd_1\lambda\ell}{2(dd_1\lambda\ell + d_2 n)}, \frac{\delta_2 d_2 n}{dd_1\lambda\ell + d_2 n}\right).$$

Proof. Codewords are of length $dd_1\lambda\ell + d_2 n$, so an adversary who can corrupt a δ fraction of the bits of can corrupt at most $\delta(dd_1\lambda\ell + d_2 n)$ bits of the entire codeword. The decoder must be able to recover, even if the adversary focuses all the errors on **SKLDC**, so we must have

$$\delta(dd_1\lambda\ell + d_2 n) \leq \delta_2 d_2 n \Rightarrow \delta \leq \frac{\delta_2 d_2 n}{dd_1\lambda\ell + d_2 n}.$$

We also need a decoder to recover sk' with all but negligible probability, even if the adversary focuses all its errors on the control portion of codeword. Recall that the control information $\mathbf{ECCEnc}_1(\mathsf{Enc}(sk', r))$ is $dd_1\lambda$ bits long, and there are ℓ copies of it in the codeword. Let Z_j denote the event that the jth control bit read by the decoder is corrupted, where the probability ranges over the decoder's choice over which of the ℓ copies the bit is read from. Then each Z_j is an independent Bernoulli random variable, and $\sum_{i=1}^{dd_1\lambda} \mathbb{E}(Z_i) = \alpha_1 dd_1\lambda$. So Chernoff tells us that

$$\Pr[\mathfrak{e}^c] = \Pr\left[\sum_j Z_j > \delta_1 dd_1\lambda\right] < \left(\frac{e^{\frac{\delta_1}{\alpha_1}-1}}{\left(\frac{\delta_1}{\alpha_1}\right)^{\frac{\delta_1}{\alpha_1}}}\right)^{2\alpha_1 dd_1\lambda}$$

Where α_1 is the fraction of the control information that is corrupted. So, even if the adversary focuses all the corruptions on the control information, we have

$$\alpha_1 \leq \frac{\delta(dd_1\lambda\ell + d_2 n)}{dd_1\lambda\ell}.$$

We observe that this will clearly be negligible in λ, whenever $\delta_1 > \alpha_1$, i.e. the error tolerance of $\mathbf{ECC}_1$ is greater than the proportion of the control information that is corrupted. By choosing ℓ to be large enough and δ to be small enough, we can always ensure that this is the case. In particular, we require

$$\delta < \frac{\delta_1 dd_1\lambda\ell}{dd_1\lambda\ell + d_2 n}.$$

Thus we have the required result whenever

$$\delta < \min\left(\frac{\delta_1 d d_1 \lambda \ell}{d d_1 \lambda \ell + d_2 n}, \frac{\delta_2 d_2 n}{d d_1 \lambda \ell + d_2 n}\right),$$

Notice that δ will be constant whenever $\lambda\ell = \mathcal{O}(n)$, which will occur if $\ell = \mathcal{O}(\lambda)$ and $n \geq \lambda^2$. □

5 Conclusion

In this work we showed how to design locally decodable codes in the computationally bounded channel model, achieving constant expansion and tolerating a constant fraction of errors, based on the existence of IND-CPA secure public-key encryption.

This is the first work giving public-key locally decodable codes in the bounded channel model with keys that are independent of the size of the message, and the only public-key locally decodable codes achieving constant rate based on standard assumptions.

Our constructions are also fairly efficient. The decoder must do a single decryption with an IND-CPA secure cryptosystem, two evaluations of PRGs, and then decode two standard error-correcting codes.

Our construction is easily modified to provide a transformation from any secret-key locally decodable code to a public-key one.

References

[BC11] Rishiraj Bhattacharyya and Sourav Chakraborty. Constant query locally decodable codes against a computationally bounded adversary. http://people.cs.uchicago.edu/ sourav/papers/LDCbounded.pdf, 2011.

[BFLS91] Laszlo Babi, Lance Fortnow, Leonid Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *STOC '91*, pages 21–31, 1991.

[CMS99] Christian Cachin, Silvio Micali, and Markus Stadler. Computationally private information retrieval with polylogarithmic communication. In *Advances in Cryptology: EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 402–414. Springer Verlag, 1999.

[Efr09] Klim Efremenko. 3-query locally decodable codes of subexponential length. In *STOC '09*, pages 39–44. ACM, 2009.

[GLD04] Parikshit Gopalan, Richard J. Lipton, and Yan Z. Ding. Error correction against computationally bounded adversaries. Manuscript, 2004.

[GS10] Venkatesan Guruswami and Adam Smith. Codes for computationally simple channels: Explicit constructions with optimal rate. In *FOCS '10*, 2010.

[HO08] Brett Hemenway and Rafail Ostrovsky. Public-key locally-decodable codes. In *CRYPTO*, pages 126–143, 2008.

[HS05] Don Hush and Clint Scovel. Concentration of the hypergeometric distribution. *Statistics and Probability Letters*, 75:127–132, 2005.

[KSY11] Swastik Kopparty, Shubhangi Saraf, and Sergey Yekhanin. High-rate codes with sublinear-time decoding. In *STOC '11*, 2011.

[KT00] Jonathan Katz and Luca Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *STOC '00: Proceedings of the 32nd Annual Symposium on the Theory of Computing*, pages 80–86, 2000.

[Lip94] Richard J. Lipton. A new approach to information theory. In *STACS '94: Proceedings of the 11th Annual Symposium on Theoretical Aspects of Computer Science*, pages 699–708, London, UK, 1994. Springer-Verlag.

[MPSW05] Silvio Micali, Chris Peikert, Madhu Sudan, and David A. Wilson. Optimal error correction against computationally bounded noise. In Joe Kilian, editor, *TCC*, volume 3378 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 2005.

[OPS07] Rafail Ostrovsky, Omkant Pandey, and Amit Sahai. Private locally decodable codes. In *ICALP '07 : Proceedings of the 34th International Colloquium on Automata, Languages and Programming*, volume 4596 of *Lecture Notes in Computer Science*, pages 387–298. Springer, 2007.

[PS94] Alexander Polishchuk and Daniel Spielman. Nearly linear size holographic proofs. In *STOC '94*, pages 194–203, 1994.

[Sha48] Claude E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–343, 623–656, 1948.

[Sud92] Madhu Sudan. *Efficient Checking of Polynomials and Proofs and the Hardness of Approximation Problems*. PhD thesis, UC Berkeley, 1992.

[Tre04] Luca Trevisan. Some applications of coding theory in computational complexity. *Quaderni di Matematica*, 13:347 – 424, 2004.

[Yek10] Sergey Yekhanin. Locally decodable codes. *Foundations and Trends in Theoretical Computer Science*, 2010.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de