Constant Query Locally Decodable Codes against a Computationally Bounded Adversary

Rishiraj Bhattacharyya [*] Sourav Chakraborty [†]

Abstract

We consider the problem of Locally Decodable Codes against an adversary that is computationally bounded in terms of running time. Many times the computational limitations of the adversary help in designing better codes. In this paper we prove (using some standard cryptographic assumptions) that in the case of constant query locally decodable codes the computational constraint does not make the adversary less powerful, unless both the sender and the receiver shares a secret key. We show, if either the receiver or the sender has a secret key (that is not known to the other) then designing LDCs against computationally bounded adversary is as hard as designing smooth codes unless some standard cryptographic assumptions are false.

[*]Indian Statistical Institute, Kolkata. Email: {rishi_r}@isical.ac.in

[†]Chennai Mathematical Institute. Email: {sourav}@cmi.ac.in. A part of the work was done when Sourav Chakraborty was at Technion, Israel Institute of Technology

1 Introduction

Modern study of error-correcting codes started with the seminal papers of Shannon [14] and Hamming [5]. The main goal is to communicate successfully over a noisy channel. Say Alice wants to send an n-bit message to Bob through a channel. Unfortunately Eve (read as nature or an evil adversary) tries to play spoilsport between Alice and Bob and adds noise to the message while it is being transmitted via the channel. To get around the problem of Eve, Alice and Bob agrees on a protocol before hand. Of course if Eve is all powerful and can add arbitrary amount of noise in the message then clearly Alice and Bob are at the mercy of Eve irrespective of how clever their protocol may be. So various reasonable restrictions on the power of Eve are usually considered. The goal of error correcting codes is to design efficiently executable protocols under various restrictions on Eve.

All the known protocols for error correction works on the simple framework of encoding and decoding. That is Alice encodes her n-bit message as a m-bit string (adding redundancy cleverly). Bob on receiving the noisy m-bit string manages to decipher the original message (cleverly using the redundant bits). The code length m is a function of the message length n. The function m is called the rate of the protocol.

Clearly the efficiency of an error correcting code (rate of the code and the error correcting ability) depends on the power of Eve. Shannon [14] in his seminal paper assumed that Eve can flip every bit of the encoded message independently with a certain probability. This is a very reasonable restriction on Eve assuming she is not malicious. But in real life it is often seen that errors can be introduced in a malicious way (like errors can be introduced in blocks as in a scratch on a compact disk). On the other hand Hamming in his seminal paper [5] assumed that Eve is allowed to flip a certain fraction of the bits in the encoded message and she can do it in an adversarial way. That is, Eve is assumed to be computationally unbounded and can find out the best way of introducing errors under the restriction that the total amount of errors is less than a certain fraction of the original message. Several other kinds of restrictions on Eve has also been considered all trying to model noise that occurs in real life [15, 20].

The Computationally Bounded Adversary model was introduced in [9]. In this model, we assume that Eve can flip a certain fraction of the bits in an adversarial way but she is also computationally bounded, that is, to decide which bits to flip Eve can only run a polytime algorithm. Thus in this model the adversary (Eve) is more powerful than its counterpart in the Shannon's Model [14] but weaker than the one in the Hamming Model [5]. Eve in this computationally bounded adversary model is also more powerful than Eves in many other models studied earlier. Taking a bit of poetic liberty one can say that nature is malicious but also computationally bounded. Thus it can be argued that the computationally bounded adversary model is an appropriate modeling of how noise is introduced in real life. In any case one central question is how powerful is Eve in the computationally bounded adversary model compared to the Eve in the Hamming model.

The art of decoding has also changed over the years. Traditionally decoding meant "unique decoding" that is, the decoder wants to decipher the whole original message. But many times the decoder is just interested in decoding only a few bits of the original message. This is modeled by Locally Decodable Codes (LDCs). A q-query LDC consists of a randomized decoder that on input i looks at only q bits of the received corrupted message and predicts the ith bit of the original input with high probability (say with probability $> 2/3$). The formal definition is given in Section 2.

LDC for the Hamming model (called Information Theoretic LDC) has received a lot of attention in the past decade but still our knowledge in this field is far from satisfactory. For $q = 2$ we have [1, 8, 16] tight exponential bound on the rate. But for $q \geq 3$ our best lower bound on the rate is less than quadratic [19] while the best upper bound is quasi-polynomial [2, 21].

In this paper we consider the problem of LDC in the computationally bounded adversary model. The formal definition is given in Section 2. Since Eve is computationally bounded it is reasonable to expect that we can design LDCs (may be using some computational hardness assumptions) that can achieve better rates than information theoretic LDCs. In [10], Micali *et al* proved that under the assumption that One way function exists (hence digital signature scheme exists) one can design a uniquely decodable code in the computationally bounded adversary model which have strictly better error correcting capacity than that of the uniquely decodable codes in the Hamming model. We try to answer the equivalent question in the LDC setting. Namely, can one design a q-query LDC (for small or constant q) in the computationally bounded adversary model which have strictly better rate that the LDCs in the Hamming Model. From our results we get a better understanding of the power of computationally bounded adversaries.

1.1 Models for LDC against Computationally Bounded Adversary

Since the adversary (Eve) is assumed to be computationally bounded, it is natural to design protocols based on certain cryptographic assumptions. The security of any such protocol is based on the fact that there is a "secret key" which the computationally bounded adversary is not expected to guess or find out. Depending on whether the secret key is generated by the sender (Alice) or the receiver (Bob) or whether the secret key is known to both, the following three models arise. We give the informal description of the three models below. The formal definitions are given in Section 2.

Digital Signature Model: In this model Alice (the sender) has a secret key (to be used for the digital signature of Alice) and she encodes the message using her secret key. All other parties (Bob and Eve) know the corresponding public key. Bob, upon receiving the corrupted codeword, uses his decoding algorithm and the public key to decode the required message bit. Micali *et al* [10] used this model to construct unique decoding codes in the computationally bounded adversary model that has strictly better error correcting rate than any code in Hamming Model.

Public Key Model: The second model is called the public key model. In this model, Bob (receiver) generates his own public key and secret key and publishes the public key. Both Alice and Eve knows the public key. Alice encodes the message using Bob's public key. Bob uses his private key to decode the message bit by querying q bits from the corrupted codewords. In [6] Hemenway and Ostrovsky considered this model and they constructed a linear rate (κ^2) query LDC under ϕ-hiding assumption, where κ is the security parameter.

Symmetric Key Model: The third model is called the symmetric key model. In this model Alice and Bob share a secret key SK, which is unknown to Eve. Alice encodes the message using SK and transmits the codeword. Bob, after receiving the corrupted codeword, decodes the required bit using SK. Clearly this model is the strongest model in terms of security and both Alice and Bob has a common secret key. In [11], Ostrovsky *et al* considered a similar model where the decoder was required to output the correct bit with probability more than $(1 - 1/poly(\kappa))$, where κ is the security parameter. But their proof can easily be extended to the model when the decoder has to output the correct message bit with probability greater than any particular constant that is more than $1/2$. Thus from [11] we have a constant query polynomial rate LDC in the Symmetric Key Model.

1.2 Main Results

In this paper we prove that in the digital signature model and the public key model one cannot obtain better LDCs than in the Hamming model (for constant queries). This is in sharp contrast to the symmetric key

model where we know there have a constant query polynomial rate LDC, which is much better than what we currently have for Hamming model.

Most of the lower bounds on LDCs were proved via smooth codes, introduced by Katz and Trevisan [7]. $C : \{0,1\}^n \to \{0,1\}^m$ is a q-query smooth code if there exists a decoding algorithm that looks at the bits at q "random" positions in the uncorrupted codeword and outputs the correct bit with high probability. Since there is no question of noise is the definition, so whether a code is a smooth code is irrespective of the model of adversary we are considering. For formal definition of smooth code see Section 2. Katz and Trevisan proved the following theorem.

Theorem 1.1 *[7] In C is a (q, δ, ϵ)-locally decodable code in the Hamming model then C is a $(q, q/\delta, \epsilon)$-smooth code.*

Because of the combinatorial nature of smooth codes it is often easier to obtain lower bounds on the rate of smooth codes. This in turn gives a lower bound for locally decodable code. On the other hand for constant q if we have a (q, c, ϵ)-smooth code then we also have a (q, δ', ϵ')-LDC in the Hamming model [17]. Thus the parameters for smooth codes are almost the best one can expect from LDCs in the Hamming model.

For our lower bound results on the rate of locally decodable codes against computationally bounded adversary in the Digital Signature Model and the Public Key Model we prove theorems analogous to Theorem 1.1 for computationally bounded adversary. Thus LDCs against computationally bounded adversaries in the Digital Signature Model and the Public Key Model are as hard as smooth codes and hence as hard as LDC against computationally unbounded adversaries. Thus in the Digital Signature Model and the Public Key Model the extra information that the adversary in computationally bounded does not help us in constructing more efficient LDCs.

In the Digital Signature Model the proof of the main theorem is very similar in flavor to the proof of Theorem 1.1. But the computational limitation of the adversary makes the proof a bit more technical.

Theorem 1.2 *For any $\delta' < \delta$ and $\rho > 0$ if C is a (q, δ, ϵ)-query LDC in the Digital Signature Model then C is a $(q, \frac{q}{\delta'} + \rho, \epsilon)$-smooth code.*

The above theorem proves that in the digital signature model one cannot obtain better LDCs than in the Hamming model (for constant queries).

Thus if one want to obtain a lower bound on the rate of a LDC in the digital signature model one only needs to obtain a lower bound on the rate of smooth code. Unfortunately our current knowledge about smooth code in general is not very good. And in many of the lower bounds for rate of smooth code it is assumed that the code has some other nice properties; usually they are assumed to be linear. A code $C : \{0,1\}^n \to \{0,1\}^m$ is linear if there exist a $n \times m$ matrix M such that for all $x \in \{0,1\}^n$, $C(x) = M.x$. In fact most of our commonly used codes are linear.

Most of the LDC that are commonly used have some extra properties. Usually they satisfy the condition that if no error is introduced then the output of the decoder is correct with probability 1. This property is known as completeness. Also, the decoder is usually linear in the sense that the decoder outputs the XOR of all the bits it queries. These are very natural property to have in a code.

In the Public Key Model to show that one cannot obtain better LDCs than in the Hamming model (for constant queries) we will need to assume that the code is linear and complete.

Theorem 1.3 *Let $q > 1$ be a fixed integer. Let $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ be a (q, δ, ϵ)-Linear Public Key LDC with full completeness and based on blackbox reduction of any underlying one way trapdoor function. Let the message length be n and the rate be $\rho(n)$. Then there exists a $(q, \frac{q^2}{\delta}, \epsilon)$-smooth code with rate $\rho(qn)$.*

For the proof of the above theorem, we show that either we can construct a smooth code from the LDC or we can construct a 1-out-of-2 Oblivious Transfer protocol which has communication less than the known lower bound and hence not possible.

Although the conditions of linearity and completeness are necessary for our proof we believe that using a slightly more careful analysis one should be able to obtain a similar result without these restrictions. We also have to assume that the protocol is based on a fully blackbox reduction of any underlying one way trapdoor function. This is because we use the lower bound result on 1-out-of-2 Oblivious Transfer protocol from Haitner *et al* [4] and their lower bound holds for protocol that are based on a fully blackbox reduction of any underlying one way trapdoor function. Thus if for some other type of protocols a similar lower bound can be achieved then our theorem will hold for those types of protocols too.

In the symmetric key model the results of Ostrovsky *et al* [11] can easily be extended to obtain a subquadratic rate constant query LDC. For completeness, we give a proof of the following theorem in the appendix.

Theorem 1.4 *There exist constant q, fixed real δ, ϵ and a (q, δ, ϵ) Symmetric Key Locally Decodable Code of rate $\mathcal{O}(n \log^2 n)$.*

1.3 Previous Results on LDC

Locally Decodable Codes were first formally introduced by Katz and Trevisan in [7]. For 2-query LDCs tight lower bound of $2^{\theta(n)}$ was given in [3] for linear code and in [8] for nonlinear codes. For $q \geq 3$, the best known lower bound of $\Omega(n^{\frac{q+1}{q-1}}/\log n)$ is due to Woodruff [19]. On the other side, in a breakthrough work [21], Yekhanin showed a construction of 3-query LDC of rate $exp\left(n^{\mathcal{O}(\frac{1}{\log \log n})}\right)$ under the assumption of existence of infinite number of Mersenne primes. In a recent work [2], Efermenko has obtained a 3-query LDC of subexponential length without the assumption.

1.4 Organization of the Paper

The rest of the paper is organized as follows: In Section 2, we introduce the notations and definitions used throughout the paper. In Section 3 we prove the lower bound of constant query LDC in Digital Signature model. For public key LDCs our proof is described in Section 4. In the Appendix 8 we present a construction of constant query non-smooth LDC under symmetric key setup.

2 Notations and Definitions

In this paper the concatenation of two vectors x and y is denoted by $x||y$. $[n]$ denotes the set $\{1, 2, ..., n\}$. A negligible function in n is one which vanishes faster than $\frac{1}{p(n)}$ for any fixed polynomial $p(n)$. We denote a negligible function by $\nu(n)$.

In the rest of the section we present the formal definitions of the various different concepts that are relevant to this paper. We start with definitions of locally decodable codes, locally decodable codes against computationally bounded adversary and smooth codes.

Definition 2.1 *[Locally Decodable Codes] For any positive integer q and reals δ, ϵ a code $C : \{0,1\}^n \to \{0,1\}^m$ is a* (q, δ, ϵ)-locally decodable code *(or (q, δ, ϵ)-LDC for short) if there is a probabilistic algorithm D such that*

- *for every $x \in \{0,1\}^n$, and any $y \in \{0,1\}^m$ with $d(C(x), y) \leq \delta m$ and for every $i \in [n]$ we have*

$$\Pr(D(i, y) = x_i) \geq \frac{1}{2} + \epsilon$$

where the probability is taken over internal random coin toss of D.

- *every invocation of $D(i, y)$ reads at most q bits of y.*

All the previous works on LDC against computationally bounded adversary are on different models and hence the definitions are different. Here we present a generalized definition of a LDC against a computationally bounded adversary. We start with the observation that, in the above definition, for every y with $d(C(x), y) \leq \delta m$, the decoding algorithm must output correct bit with probability more than $\frac{1}{2} + \epsilon$. Now a polynomial time adversary is allowed to use random coins. So it is possible that with a very small probability the computational adversary is acting like a computationally unbounded adversary. So if we hope to use the computational limitations of the adversary to design better codes then we have to allow the decoder to be completely wrong with a very small probability. So it is natural to consider keyed LDC to get some better rate. Now if y is the corrupted message then with probability $1 - \nu(\kappa)$ (probability is taken over joint distribution of randomness of key generation algorithm and the adversary) the decoder has to correctly output the ith bit with probability more than $\frac{1}{2} + \epsilon$ (here the probability is taken over decoders random coin tosses) more than $\frac{1}{2} + \epsilon$). Here $\nu(\kappa)$ is a negligible function of security parameter κ.

Definition 2.2 *[LDC against a computationally bounded adversary] For any positive integer q and reals δ, ϵ and any security parameter κ a code $C : \{0,1\}^n \to \{0,1\}^m$ is a* $(q, \delta, \epsilon, \kappa)$-Locally decodable code against a computationally bounded adversary *if there is a probabilistic decoding algorithm D such that for any adversary $\mathcal{A} : \{0,1\}^m \to \{0,1\}^m \times [n]$ whose running time is bounded by a polynomial in κ we have*

- *for every $x \in \{0,1\}^n$, and $\mathcal{A}(C(x)) = (y, i)$ if $d(C(x), y) < \delta m$ then*

$$\Pr_{\mathcal{A}}\left[\Pr_{D}(D(i, y) = x_i) \geq \frac{1}{2} + \epsilon\right] > 1 - \nu(\kappa)$$

where the first probability is taken over coin tosses of $\mathcal{A}$ and second probability is taken over internal random coin toss of D.

- *every invocation of $D(i, y)$ reads at most q bits of y.*

Definition 2.3 *[Smooth Codes] For any positive integer q and reals c, ϵ a code $C : \{0,1\}^n \to \{0,1\}^m$ is a* (q, c, ϵ)-smooth code *if there is a probabilistic algorithm D such that*

- *for every $x \in \{0,1\}^n$ and for every $i \in [n]$*

$$\Pr[D(i, C(x)) = x_i] \geq \frac{1}{2} + \epsilon,$$

- *for every $i \in [n]$ and every $j \in [m]$*

$$\Pr[D(i, C(x)) \text{ queries bit } j] \leq \frac{c}{m},$$

- *every invocation $D(i, C(x))$ reads at most q bits of $C(x)$*

where the probabilities are taken over the random coin tosses of D.

Now we give the formal definitions of the three different models for constructing LDCs against computationally bounded adversary.

Definition 2.4 *[Locally Decodable Codes in Digital Signature Model]For any positive integer q and positive real numbers δ, ϵ a (q, δ, ϵ)-*Digital Signature Locally Decodable Code *(or (q, δ, ϵ)-DSLDC in short) is a triple of probabilistic polynomial time algorithms $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ such that for any positive κ (security parameter) and for all large enough n (message length)*

- $(PK, SK) \leftarrow \mathcal{G}(1^\kappa)$*;*
- $c \leftarrow \mathcal{E}(x, SK)$ *where $x \in \{0,1\}^n$ and $c \in \{0,1\}^m$;*
- $b \leftarrow \mathcal{D}(i, c', PK)$ *where $c' \in \{0,1\}^m$ and $i \in [n]$ and D queries at most q bits of c';*

such that for all for any polynomial-time adversary A_1 and A_2

$$\Pr[(PK, SK) \leftarrow \mathcal{G}(1^n); (x, \gamma) \leftarrow A_1(PK);$$
$$c \leftarrow \mathcal{E}(x, SK); (c', i) \leftarrow A_2(c, \gamma) : \Pr[\mathcal{D}(i, c', PK) = x_i] > \frac{1}{2} + \epsilon] > 1 - \nu(\kappa).$$

where $x \in \{0,1\}^n$ and $d(c, c') < \delta m$ and the first probability is taken over all internal coin tosses of $\mathcal{G}, \mathcal{E}$,, A_1 and A_2 and the second probability is over internal coin tosses of $\mathcal{D}$.

Definition 2.5 *[Public Key Locally Decodable Codes]For any positive integer q and positive real numbers δ and ϵ a (q, δ, ϵ)-*Public Key Locally Decodable Code *(or (q, δ, ϵ)-PKLDC in short) is a triple of probabilistic polynomial time algorithms $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ such that for positive κ (security parameter) and for all large enough n (message length)*

- $(PK, SK) \leftarrow \mathcal{G}(1^\kappa)$*;*
- $c \leftarrow \mathcal{E}(x, PK)$ *where $x \in \{0,1\}^n$ and $c \in \{0,1\}^m$;*
- $b \leftarrow \mathcal{D}(i, c', SK)$ *where $c' \in \{0,1\}^m$ and $i \in [n]$ and D queries at most q bits of c';*

such that for all for any polynomial-time adversary A_1 and A_2

$$\Pr[(PK, SK) \leftarrow \mathcal{G}(1^n); (x, \gamma) \leftarrow A_1(PK);$$
$$c \leftarrow \mathcal{E}(x, PK); (c', i) \leftarrow A_2(c, \gamma) : \Pr[\mathcal{D}(i, c', SK) = x_i] > \frac{1}{2} + \epsilon] > 1 - \nu(\kappa)$$

where $x \in \{0,1\}^n$ and $d(c, c') < \delta m$ and the first probability is taken over all internal coin tosses of $\mathcal{G}, \mathcal{E}$, A_1 and A_2 and the second probability is over internal coin tosses of $\mathcal{D}$.

$(\mathcal{G}, \mathcal{E}, \mathcal{D})$ is a Linear PKLDC $\mathcal{D}$ only outputs the XOR of the bits that it queries. That is $\mathcal{D}$ can only decide what all positions to query and once it queries those positions it outputs the XOR of the bits.

We say $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ is a (q, δ, ϵ)-PKLDC with full completeness if for all $x \in \{0,1\}^n$ and $i \in [n]$

$$\Pr[(PK, SK) \leftarrow \mathcal{G}(1^n); c \leftarrow \mathcal{E}(x, PK) : \mathcal{D}(i, c, SK) = x_i] = 1$$

that is, if the adversary does not corrupt the codeword then the decoder outputs the correct answer with probability 1.

Definition 2.6 *[Symmetric Key Locally Decodable Codes] For any positive integer q and positive real numbers $delta$ and ϵ a* (q, δ, ϵ)-symmetric key locally decodable code *(or (q, δ, ϵ)-SKLDC in short) is a triple of probabilistic polynomial time algorithms $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ such that for positive κ (security parameter) and for all large enough n*

- $(SK) \leftarrow \mathcal{G}(1^\kappa)$*;*
- $c \leftarrow \mathcal{E}(x, SK)$ *where $x \in \{0,1\}^n$ and $c \in \{0,1\}^m$;*
- $b \leftarrow \mathcal{D}(i, c', SK)$ *where $c' \in \{0,1\}^m$ and $i \in [n]$ and D queries at most q bits of c';*

such that for all for any polynomial-time adversary A_1 and A_2

$$\Pr[(SK) \leftarrow \mathcal{G}(1^n); (x, \gamma) \leftarrow A_1(.);$$
$$c \leftarrow \mathcal{E}(x, SK); (c', i) \leftarrow A_2(c, \gamma) : \Pr[\mathcal{D}(i, c', SK) = x_i] > \frac{1}{2} + \epsilon] > 1 - \nu(\kappa)$$

where $x \in \{0,1\}^n$ and $d(c, c') < \delta m$ and the first probability is taken over all internal coin tosses of $\mathcal{G}, \mathcal{E}$, A_1 and A_2 and the second probability is over internal coin tosses of $\mathcal{D}$.

2.1 Blackbox Reduction

A reduction of a primitive P to a primitive Q is a construction of P from Q. Such a construction consists of showing that if there exists an implementation $\mathcal{C}$ of Q, then there exists an implementation $\mathcal{M}_\mathcal{C}$ of P. This is equivalent to showing that for every adversary that breaks $\mathcal{M}_\mathcal{C}$, there exists an adversary that breaks C. Such a reduction is semi-blackbox if it ignores the internal structure of Qs implementation, and it is fully blackbox, if the adversary for breaking $\mathcal{C}$ ignores the internal structure of both $\mathcal{C}$ and of the (assumed) adversary breaking $\mathcal{M}_\mathcal{C}$.

2.2 $\binom{2}{1}$ Oblivious Transfer

A 1-out-of-2 Oblivious Transfer ($\binom{2}{1}$ OT) is a protocol between two communicating parties; one sender and one receiver. The sender has two secret bits b_0 and b_1 and the receiver's input is value $\sigma \in \{0,1\}$. At the end of the protocol the receiver retrieves b_σ while the sender does not learn σ. The security of the protocol guarantees that a cheating receiver should not be able to learn the bit $b_{1-\sigma}$ whereas a cheating sender should not know σ. A $\binom{2}{1}$ OT is called statistically-sender-private if the view of the receiver is statistically close for both possibilities of the unrevealed bit.

In [4] Haitner *et al* proved the following Theorem

Theorem 2.7 *Any $O(\kappa)$-security-parameter-expanding fully-black-box construction of a $\binom{2}{1}$ statistically sender private OT protocol has $\Omega(\frac{\kappa}{\log \kappa})$ rounds.*

As in every round, the sender must send at least one bit, we get the immediate corollary

Corollary 2.8 *In any $O(\kappa)$-security-parameter-expanding fully-black-box construction of a $\binom{2}{1}$ statistically sender private OT protocol, the sender sends $\Omega(\frac{\kappa}{\log \kappa})$ bits.*

3 LDC in Digital Signature Model

3.1 Proof of Theorem 1.2

Let us begin by recalling the proof of Katz and Trevisan [7] that is $C : \{0,1\}^n \to \{0,1\}^m$ is a (q, δ, ϵ)-ldc then C is also a $(q, q/\delta, \epsilon)$-smooth code.

Given any q-query locally decoding algorithm D for C that errs with probability at most ϵ one can prove that D can also be changed such that for all $i \in [n]$ and $j \in [m]$ $\Pr[D(i,.) \text{ queries } j] \leq q/\delta m$. Let

$$Bad_\delta(i) = \{j \mid \Pr[D(i,.) \text{ queries } j] > q/\delta m\}.$$

As D make at most q queries so $|Bad_\delta(i)| \leq \delta m$. Since the adversary knows the decoding algorithm so she also knows the set $Bad_\delta(i)$. Assume that the adversary makes all the bits corresponding to the indices in $Bad_\delta(i)$ have value 0, while leaving the rest of the bits untouched. So the adversary introduces at most δm errors. Since the the code is supposed to handle any δm errors the decoding algorithm D makes at most q queries and outputs the ith bit with probability more than $1/2 + \epsilon$. Hence if a decoder D' on receiving an uncorrupted message and bit i simulates the algorithm $D(i,.)$ but only queries the bits that are not in $Bad_\delta(i)$ and assumes that the bits in $Bad_\delta(i)$ are all 0 then D' will output the correct value with probability at at least $(1/2 + \epsilon)$. Also since D' does not query bits in $Bad_\delta(i)$ it is a $(q, q/\delta, \epsilon)$-smooth decoder and hence C is a $(q, q/\delta, \epsilon)$-smooth code.

We would like to prove a similar result for the the LDC in the Digital Signature Model. In the above the proof the decoding algorithm D was expected to output the correct value when the adversary corrupts the bits in $Bad_\delta(i)$ - it is because the decoding algorithm was supposed to work against any adversary possibly computationally unbounded. But a computationally bounded adversary might not be able to figure out the indices that are in $Bad_\delta(i)$. Hence a decoding algorithm that works against a computationally bounded adversary might not be able to decoded correctly if the bits in $Bad_\delta(i)$ are corrupted. Hence the argument of Katz and Trevisan [7] does not exactly work for the computationally bounded adversaries.

But in the digital signature model since the decoder does not have a secret key, so the adversary knows the decoding algorithm (except for the outcome of the random coin tosses that the decoder might do). So while a computationally bounded adversary can run the decoding algorithm using some random coins and can find a set of indices S_i that with high probability will contain $Bad_{\delta'}(i)$ for some δ'. And if $|S_i|$ is is less than δm then the proof of Theorem 1.2 follows in the same way as Katz and Trevisan's proof.

We now give the formal proof that a computationally bounded adversary can come up with such a set S_i. Let $\delta' < \delta$. Let C be the $(q, \delta, \epsilon, \kappa)$-LDC in the digital signature model. Let $\mathcal{D}$ be the local decoding algorithm with the public key PK. Now let

$$Bad_{\delta'}(i) = \left\{j \mid \Pr[\mathcal{D}(i,.,PK) \text{ queries } j] > \frac{q}{\delta'}\right\}$$

While,

$$Bad_{\delta}(i) = \left\{j \mid \Pr[\mathcal{D}(i,.,PK) \text{ queries } j] > \frac{q}{\delta}\right\}$$

So clearly $Bad_{\delta'}(i) \subseteq Bad_\delta(i)$. Let $\frac{q}{\delta'} = \frac{q}{\delta}(1+\gamma)$ and $\mathcal{R}$ be the set of all possible outcomes of the random coin tosses made by the decoding algorithm. For any $r \in \mathcal{R}$ let $Q(r,i)$ be the set of queries made by the decoding algorithm for decoding the bit i.

On receiving the encoded word $C(x)$ and index i the adversary picks a set U of k (k to be around $O(m(\log m)\delta^2/q^2\gamma^2)$) random elements from $\mathcal{R}$. Let $Q(U,i)$ be the multi-set $\bigcup_{r\in U} Q(r,i)$. Let $n_i(j)$ denotes the number of times j occurs in the multi-set $Q(U,i)$. Let

$$D_i = \left\{ j \mid \frac{n_i(j)}{k} > \frac{q}{\delta}(1 + \frac{\gamma}{2}) \right\}$$

Lemma 3.1 *If $k = O(m(\log m)\delta^2/q^2\gamma^2)$ then for any $\rho > 0$ and large enough m with probability at least $(1 - (\rho/m))$ we have*

$$Bad_{\delta'}(i) \subseteq D_i \subseteq Bad_{\delta}(i)$$

The proof of Lemma 3.1 is given in the Appendix.

Note that $|D_i| < \delta m$. So the computationally bounded adversary can set all the bits corresponding to the set D_i to 0 while leaving the rest unchanged. Now since a computationally bounded adversary has corrupted only δ fraction of the message so the decoding algorithm $\mathcal{D}$ must be able to output the ith bit with success probability at least $(1/2 + \epsilon)$.

We now design the smooth decoding algorithm: after receiving the non-corrupted codeword and index i the decoder runs the decoding algorithm $\mathcal{D}$ in a black-box manner by reading indices as requested by $\mathcal{D}$ except for the indices in D_i (it return 0 without reading it). The new decoding algorithm is correct with probability $(1/2 + \epsilon)$. Also with probability more than $(1 - (\rho/m))$ for all i the new decoding algorithm does not query indices in $Bad_{\delta'}(i)$. So the new decoding algorithm is $(q, q/\delta' + \rho, \epsilon)$-smooth.

4 Locally decodable codes against computationally bounded adversary in the Public Key - Private Key Setting

4.1 Proof of Theorem 1.3

Let $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ be a (q, δ, ϵ)-Linear PKLDC with full completeness and based on black-box reduction of any underlying one-way trap door function. Let the rate be ρ, where ρ is a function of the message length. We want to show that if q is a constant then either there exists a $(q, \frac{q^2}{\delta}, \epsilon)$-smooth code with rate ρ' (where, $\rho'(n/2) = \rho(n)$) or there exists a computationally bounded adversary that can change δ fraction of the bits and force the decoder to make mistake with high probability.

Let us consider one round of the protocol. Let $\mathcal{G}(1^\kappa) = (PK, SK)$. In the rest of the proof we will assume that the public-key (PK) and the secret-key (SK) are fixed and the secret-key is known only to decoder while the public-key is known to all.

Now we assume that the decoder is linear. That is, on input i the decoder decides (using the secret key) to query some bits of the received message and outputs based on the XOR of these bits. Thus if y is the received word and q_i is the set of indices that decoder decides to query then the output of the decoder is $b(i, SK) \oplus \bigoplus_{j \in q_i} y_j$ where $b(i, SK)$ is any bit depending independent of the results of the queries.

To make the presentation of the proof simple we will assume that the bit $b(i, SK)$ is 0 for all i and any secret key SK. The theorem still holds without this assumption and proof is just a little bit more messy. We skip the details in this extended abstract. Thus for the rest of the prove the decoder just outputs the XOR of the bits it queries.

Definition 4.1 *For the decoding algorithm $\mathcal{D}$ the recovery graph for any index $i \in [n]$ and secret key SK is a q-regular hyper-graph $G^i(SK)$ such that*

- *$G^i(SK)$ has m vertices representing m bits of the codewords.*
- *$(i_1, i_2, ..., i_q)$ is an hyperedge in $G^i(SK)$ if $\mathcal{D}(i, ., SK)$ queries the tuple $(i_1, i_2, ..., i_q)$.*

Note that since $\mathcal{D}$ has full completeness and the output is the XOR of the bits it queries so for any $x \in \{0,1\}^n$ and any $i \in [n]$ if $(i_1, i_2, ..., i_q)$ is an hyperedge in $G^i(SK)$ then $(C(x)_{i_1} \oplus \cdots \oplus C(x)_{i_q}) = x_i$. We relax the definition of recovery graphs slightly as follows.

A family $\mathcal{H}(SK) = \{H^1(SK), \ldots, H^n(SK)\}$ of q-regular graphs on m vertices is an almost-recovery graph family if

- for all $i \in [n]$, the recovery graph $G^i(SK)$ is a subgraph of $H^i(SK)$.
- there is a $R \subset \{0,1\}^n$, with $|R| > 2^{n-1}$, such that for all $i \in [n]$ and all $x \in R$ if $(i_1, i_2, ..., i_q)$ is an hyperedge in $H^i(SK)$ then
$$(C(x)_{i_1} \oplus \cdots \oplus C(x)_{i_q}) = x_i.$$

We call an $i \in [n]$ a vulnerable index with respect to an almost-recovery graph family $\mathcal{H}$ if the hypergraph H^i has a vertex cover of size less than $\frac{\delta m}{q}$.

The main idea of the rest of the proof is that if the adversary can somehow construct an almost-recovery graph H^i for any vulnerable index i then the adversary can easily find a small vertex cover of the graph H^i and flips the bits of the vertex cover uniformly at random. Now since the decoder is linear the decoder would err will probability $1/2$ irrespective of what query set it chooses from the recovery graph G^i. The next two lemmas proves that the adversary can easily construct an almost-recovery graph H^i for any vulnerable index i unless the code is a smooth code.

Lemma 4.2 *If for any family $\mathcal{H}$ of almost-recovery graphs the number of vulnerable bits is at most $n/2$; then there exists a $(q, \frac{q^2}{\delta}, \epsilon)$-smooth code of rate ρ', where $\rho'(n/6) = \rho(n)$.*

Lemma 4.3 *There exists a polytime algorithm $\mathcal{A}$ that on input PK with high probability can construct an almost-recovery graph family $\mathcal{H}$.*

Once the adversary uses the Lemma 4.3 to construct an almost-recovery graph family $\mathcal{H}$. she picks $H^i(SK)$ for some random index i. Using Lemma 7.1 she can find an set of vertices that is an approximate vertex cover vertex cover for $H^i(SK)$.

From Lemma 4.2 we can assume that more than more that $n/2$ indices are vulnerable. Now if at least half of the indices are vulnerable w.r.t $\mathcal{H}$. the adversary can, with high probability, find an i such that $H^i(SK)$ has a vertex cover of size is at most δm (because for any vulnerable i the size of the minimum vertex cover of $H^i(SK)$ is $\frac{\delta}{q}m$). Let this set of vertices in the vertex cover be N.

Now the adversary will flip the bits of the codeword which are at positions corresponding to the set N and the resulting corrupted codeword be y'. Since the decoding algorithm $\mathcal{D}(.,.,SK)$ is a linear function,

$$Pr[\mathcal{D}(i, y', SK) = x_i] = 1/2.$$

This contradicts the condition for $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ that the decoder outputs the correct message bit with probability $1/2 + \epsilon$.

4.2 Proof of Lemma 4.3

Let $Q(i)$ be the set of the query tuples made by $\mathcal{D}(i,.,SK)$. In other words $Q(i)$ denotes the set of edges of the recovery graph of $G^i(SK)$. Since $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ is a (q, δ, ϵ)-Linear PKLDC with full completeness so for any $i, j \in [n]$, $Q_i \cap Q_j = \emptyset$.

Lemma 4.4 *Let q be a constant positive integer. Let $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ be a (q, δ, ϵ)-Linear PKLDC with full completeness and based on black-box reduction of any underlying one-way trapdoor function. Let the security parameter be $O(\kappa)$ and the message size be n. Then for all $i, j \in [n]$ there exist efficient algorithm $A_{i,j}$ and polynomial p such that with all but negligible probability over SK, we have for all but finite number of κ*

$$|Pr[x \in Q(i) : A_{i,j}(x, i) = 1)] - Pr[x \in Q(j) : A_{i,j}(x, i) = 1)]| > 1/p(\kappa).$$

The proof of Lemma 4.4 is given in the Appendix. Then by using the algorithms A_{ij} (from Theorem 4.4) repeatedly for a polynomial number of times the adversary can construct the sets $S_i^{i,j}, S_j^{i,j} \subset [m]^q$ such that with high probability $Q_i \subset S_i$ and $Q_i \subset S_j$. Now the adversary repeats this for different all pairs i, j. Let for all i, $S_i = \cap_j S_i^{i,j}$. With high probability S_i contains Q_i for all i.

Now since the adversary knows the the public key PK she can encode any random message strings $x \in \{0,1\}^n$. Now if $(i_1, \ldots, i_q)$ is in S_i and $(C(x)_{i_1} \oplus \cdots \oplus C(x)_{i_q}) \neq x_i$ then the adversary removes the tuple $(i_1, \ldots, i_q)$ from the set S_i.

Note that if $(i_1, \ldots, i_q)$ is in Q_i then $(C(x)_{i_1} \oplus \cdots \oplus C(x)_{i_q})$ is always x_i as the decoder has full completeness.

Thus by repeating the above trial polynomial number of times for all i and all elements in S_i the adversary finally obtains a family of sets $\{H_1, \ldots, H_n\}$ such that with probability

- for all i, $Q_i \in H_i$, and
- with high probability, for at least half the message strings of length n and for all i if $(i_1, \ldots, i_q)$ is in H_i and $(C(x)_{i_1} \oplus \cdots \oplus C(x)_{i_q}) = x_i$

Thus this family gives a almost-recovery graph family.

4.3 Proof of Lemma 4.2

We first need to define a slight variant of smooth codes.

Definition 4.5 *[Almost Smooth Codes] For any positive integer q and reals c, ϵ a code $C : \{0,1\}^n \to \{0,1\}^m$ is a* (q, c, ϵ)-almost-smooth code *if there is a probabilistic algorithm D such that*

- *there is a set $S \subset \{0,1\}^n$, with $|S| \geq 2^{n-1}$ such that for all $x \in S$ and for every $i \in [n]$*

$$\Pr[D(i, C(x)) = x_i] \geq \frac{1}{2} + \epsilon,$$

- *for every $i \in [n]$ and every $j \in [m]$*

$$\Pr[D(i, C(x)) \text{ queries bit } j] \leq \frac{c}{m},$$

- *every invocation $D(i, C(x))$ reads at most q bits of $C(x)$*

where the probabilities are taken over the random coin tosses of D.

Lemma 4.6 *If $C : \{0,1\}^n \to \{0,1\}^m$ is a almost-smooth code then there exists a smooth code $C' : \{0,1\}^{n/3} \to \{0,1\}^m$.*

Proof. [of 4.6] Let $S \subset \{0,1\}^n$ be the set of x for which there is a guarantee that $\Pr[D(i, C(x)) = x_i] \geq \frac{1}{2} + \epsilon$.

Now the VC-dimension of any set $S \subset \{0,1\}^n$ is the maximal k such that for some distinct $i_1, \ldots, i_k$ in $[n]$ the restriction of S to $\{i_1, \ldots, i_k\}$ contains all 2^k possible assignments. The following lemma known as Sauer's lemma was proved independently by Sauer[12], Perles and Shelah[13] and Vapnik and Chervonenkis[18]:

Lemma 4.7 *For every n and every set $S \subset \{0,1\}^n$ with VC-dimension k*

$$|A| \leq \sum_{i=1}^{k} \binom{n}{i}$$

Thus in other words if S has size more than 2^{n-1} then the VC-dimension of S is at least $n/3$, that is, there exist $i_1, \ldots, i_n/3$ such that for all $y \in \{0,1\}^{n/3}$ there exist $x(y) \in S$ such that for all $j \in [n/3]$ we have $y_j = x(y)_{i_j}$.

Now the encoding C' will take y to the respective $x(y)$ and encode $x(y)$ using C. When the decoder has to decode the jth bit will invoke the decoder for C for the i_jth bit. Thus the decoder will be smooth. Hence C' is a smooth code. ■

Now we return to the proof the Lemma 4.2. We again start with a definition.

Definition 4.8 *For any index $i \in [n]$ we say a $\mathcal{D}(.,.,SK)$ is c-smooth for i if*

$$Prob[\mathcal{D}(i,.,SK) \textit{ queries bit } j] \leq \frac{c}{m}.$$

Let $\mathcal{H}$ be the almost-recovery graph family. Firstly of all we prove that there exists a decoding algorithm $\mathcal{D}'$ such that if i is not a vulnerable index w.r.t $\mathcal{H}$ then $\mathcal{D}'$ is $\frac{q^2}{\delta}$-smooth for i.

If i is not a vulnerable index w.r.t $\mathcal{H}$ then the minimum vertex cover of $H^i(SK)$ is of size at least $\frac{\delta}{q}m$. Hence Lemma 7.1 says $H^i(SK)$ has a maximum matching M_i of size at least $\frac{\delta m}{q^2}$. We construct a decoding algorithm $\mathcal{D}'$ such that for each i, $\mathcal{D}'(i,.)$ randomly pick a hyper-edge $e \in M_i$ and queries all the bits corresponding to the edge e and invokes $\mathcal{D}(i,.,SK)$ when it queries the edge e. Since $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ has full completeness so $\mathcal{D}'(i,x) = x_i$. Also since only the bits of the edges in M_i are queried and $\mathcal{D}'$ chooses e uniformly at random from M_i so

$$prob[\mathcal{D}'(i,.,SK) \text{ queries bit } j] \leq \frac{1}{|M_i|} \leq \frac{q^2}{\delta m} \tag{1}$$

Hence if i is not a vulnerable index w.r.t $\mathcal{H}$ then $\mathcal{D}'$ is $\frac{q^2}{\delta}$-smooth for i. So if the number of vulnerable index for $\mathcal{D}$ is at most $n/2$ then number of smooth index is at least $n/2$.

Now we prove that if $\mathcal{D}'$ is $\frac{q^2}{\delta}$-smooth for more than $n/2$ indices then we can use the code $\mathcal{E}$ to construct a $(q, \frac{q^2}{\delta}, \epsilon)$-smooth code with rate ρ', such that $\rho'(n/2) = \rho(n)$.

First can define a one-one function $\chi : [\frac{n}{2}] \to [n]$ such that for all $i \in [\frac{n}{2}]$, $\mathcal{D}(.,.,SK)$ is $\frac{q^2}{\delta}$-smooth for $\chi(i)$. For $y \in \{0,1\}^{n/2}$ let $E(y) \in \{0,1\}^n$ be defined as: if $E(y)_j = y_i$ if $j = \chi(i)$ and 0 otherwise. Now let $C'(y) = C(E(y))$. Consider the decoder $\hat{D} : [\frac{n}{2}] \times \{0,1\}^m$ defined as $\hat{D}(i,y) = D'(\chi(i), y)$.

Now since we started with an almost-recovery family the code C' is actually a $(q, \frac{q^2}{\delta}, \epsilon)$-almost-smooth code with rate ρ', where $\rho'(n/2) = \rho(n)$.

But from Lemma 4.6 we obtain a $(q, \frac{q^2}{\delta}, \epsilon)$-almost-smooth code with rate ρ'', where $\rho''(n/3) = \rho'(n)$.

Acknowledgment

We would like to thank Satya Lokam, Eyal Kushilevitz, Palash Sarkar and Iftach Haitner for useful discussions. We would also like to specially thank Satya Lokam for introducing us to the subject of error correcting against computationally bounded adversary.

References

[1] Avraham Ben-Aroya, Oded Regev and Ronald de Wolf. A Hypercontractive Inequality for Matrix-Valued Functions with Applications to Quantum Computing and LDCs. In FOCS 08: 49th Annual Symposium on Foundations of Computer Science, pages 477-486,2008.

[2] Klim Efremenko. 3-query locally decodable codes of subexponential length In *STOC 09: Proceedings of the 41st ACM Symposiom on the Theory of Computing*, pages 39-44, 2009

[3] Oded Goldreich and Howard J. Karloff and Leonard J. Schulman and Luca Trevisan Lower Bounds for Linear Locally Decodable Codes and Private Information Retrieval In *CCC 02:Proceedings of Conference on Computational Complexity*, page 175-183, 2002

[4] Iftach Haitner, Jonathan Hoch, Omer Reingold and Gil Segev. Finding Collisions in Interactive Protocols - A Tight Lower Bound on the Round Complexity of Statistically-Hiding Commitments. In *FOCS 07: Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, pages 669-679, 2007

[5] R. W. Hamming. Error detecting and error correcting codes. In *Bell Systems Technical Journal*, 29, pages 147-150, 1950.

[6] Brett Hemenway and Rafail Ostrovsky. Public-Key Locally-Decodable Codes In *CRYPTO 08*, pages 126-143.

[7] 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.

[8] Iordanis Kerenidis and Ronald de Wolf. Exponential lower bound for 2-query locally decodable codes via a quantumn argument. In *STOC 03: Proceedings of the 35th Annual Symposium on Theory of Computing*, pages 106-115, 2003

[9] 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, Springer-Verlag, 1994.

[10] Silvio Micali, Chris Peikert, Madhu Sudan, and David A. Wilson. Optimal error correction against computationally bounded noise In *TCC 05: Proceedings of the 2nd Theory of Cryptography Conference*, pages 1-16, Springer-Verlag, 2005.

[11] 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*, pages 387-398, Springer-Verlag, 2007.

[12] N. Sauer. On the density of families of sets. In *Journal of Combinatorial Theory, Series A 13* pages 145-147, 1972.

[13] S. Shelah. A combinatorial problem; stability and order for models and theories in infinitary languages. In *Pacific Journal of Mathematics,41*, pages 247261, 1972.

[14] Claude E. Shannon. A mathematical theory of communication. In *Bell System Technical Journal, 27* pages 379-343, 623-656, 1948.

[15] Claude E. Shannon. The zero error capacity of a noisy channel. IEEE Trans. Inform. Theory, Vol. IT-2, no. 3, (1956): 8-19. (Reprinted in D. Slepian, Ed., Key Papers in the Development of Information Theory. New York: IEEE Press (1974): 112-123)

[16] Dungjade Shiowattana, Satyanarayana V. Lokam. An optimal lower bound for 2-query locally decodable linear codes. Inf. Process. Lett. 97(6): 244-250 (2006)

[17] Luca Trevisan. Some Application of Coding Theory in Computational Complexity. In *Quaderni di Mathematica 13* pages 347-424, 2004.

[18] V.N. Vapnik and A.Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. In *Theory of Probability Applications, 16*, pages 264-280, 1971.

[19] D. Woodruff. New Lower Bounds for General Locally Decodable Codes. In *Electronic Colloquium on Computational Complexity (ECCC), vol 14, no 6*, 2007.

[20] Sebasti Xamb-Descamps Block Error-Correcting Codes: A Computational Primer (Universitext). Publishers: Springer.

[21] Sergey Yekhanin. Towards 3-query locally decodable codes of subexponential length. In *STOC 07: Proceedings of the 39th ACM Symposiom on the Theory of Computing*, pages 266-274, 2007

Appendix

5 Proof of Lemma 3.1

Let $p_i(j)$ denote the $\Pr[\mathcal{D}(i,.)$ queries $j]$. So if $j \in Bad_{\delta'}(i)$ then $p_i(j) > q(1+\gamma)/\delta m$. Now

$$\Pr\left[j \notin D_i\right] = \Pr\left[\frac{n_i(j)}{k} < \frac{q}{\delta}(1+\frac{\gamma}{2})\right] = \Pr\left[(\frac{qk}{\delta m}(1+\gamma) - n_i(j) > k\frac{qk}{\delta m}.\frac{\gamma}{2}\right]$$

If $j \in Bad_{\delta'}(i)$ then $p_i(j) > q(1+\gamma)/\delta m$ and hence

$$\Pr\left[j \notin D_i\right] < \Pr\left[p_i(j)k - n_i(j) > k\frac{qk}{\delta m}.\frac{\gamma}{2}\right]$$

Since $E[n_i(j)] = kp_i(j)$ by Chernoff Bound

$$\Pr[j \notin D_i] < \exp(-\frac{q^2\gamma^2k^2}{2\delta^2 m})$$

Thus if $k = O(m(\log m)\delta^2/q^2\gamma^2)$

$$\Pr[j \notin D_i] < \exp(-m)$$

Hence by union bound it follows that with probability greater than $(1 - o(1/m))$ for all i, $Bad_{\delta'}(i) \subseteq D_i$.

Similarly the other direction follows, that is, with probability greater than $(1 - o(1/m))$ for all i, $D_i \subseteq Bad_{\delta}(i)$.

Thus with probability at least $(1 - o(1/m))$ we have

$$Bad_{\delta'}(i) \subseteq D_i \subseteq Bad_{\delta}(i)$$

and for large enough m the success probability is greater than $(1 - (\rho/m))$ for any given $\rho > 0$.

6 Proof of Lemma 4.4

Let $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ be a (q, δ, ϵ)-Linear PKLDC with full completeness and based on black-box reduction of any underlying one-way trap door function. Let the message size be n and the codeword size is m. Let $Q(i)$ be the set of the query tuples made by $\mathcal{D}(i, ., SK)$, where $(PK, SK) = \mathcal{G}(r, 1^\kappa)$. Because of the linearity of the decoding algorithm $\mathcal{D}$, for $i, j \in [n]$, $Q(i) \cap Q(j) = \emptyset$. Our arguments can be extended easily to nonlinear case also with a little messier proof.

Suppose for very high fraction of r there exist indices $i, j \in [n]$ such that for all probabilistic polynomial time algorithm A

$$|Pr[x \leftarrow Q(i) : A(x, PK) = 1] - Pr[x \leftarrow Q(j) : A(x, PK) = 1]| < \nu(\kappa) \tag{2}$$

where probability is taken over random coin tosses of A and $\nu(\kappa)$ is negligible in security parameter κ. Then we show that there exist a $\binom{2}{1}$ Oblivious Transfer protocol where the sender sends q bits.Recall that, in oblivious transfer protocol the receiver wants to know 1 out of 2 secret bits (b_0 and b_1) known to the sender. The protocol is correct if, after running the protocol the receiver knows the desired bit $b_\sigma(\sigma \in \{0, 1\})$. The security of an oblivious transfer protocol requires that the receiver knows only b_σ and the server should be "oblivious" about the value σ.

The protocol works as following

1. Sender's input b_0 and b_1.
2. Receiver's input $\sigma \in \{0, 1\}$.
3. Receiver runs $\mathcal{G}(1^\kappa)$ and gets (PK, SK).
4. Receiver sends (PK, i, j, w) where $i, j \in [n]$ satisfies Equation 2 and w is any element of $Q(\phi(\sigma))$, with $\phi(0) = i$ and $\phi(1) = j$.
5. Sender constructs a n bit string x such that $x_i = b_0, x_j = b_1$ and rest of the bits are filled randomly.
6. Sender runs $\mathcal{E}(x, PK)$.
7. Sender sends the bits of $\mathcal{E}(x, PK)$ queried in w. Let the response is y.
8. Receiver on receiving the bits run $\mathcal{D}(\phi(\sigma), y, SK)$. and gets $x_{\phi(\sigma)}$.

The correctness of above protocol follows from the correctness of $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ as a Linear PKLDC with full completeness. As $Q(i, r) \cap Q(j, r) = \emptyset$, we have $w \notin Q(\phi(1 - \sigma))$. Using the linearity of $(\mathcal{G}, \mathcal{E}, \mathcal{D})$, we get

$$Pr[x_{\phi(1-\sigma)} = 0|y] = Pr[x_{\phi(1-\sigma)} = 1|y] \tag{3}$$

where the probability is taken over randomness of x. Hence the protocol is statistically sender private. The security of the receiver is implied from Equation 2.

Remark: Our arguments cover the cases when a queried tuple w from $Q(i)$ contains codeword bit(s) that is xor two bits x_i and x_j. As in that case, reconstructing x_i would immediately imply reconstruction of x_j resulting $w \in Q(j)$. That would contradict the linearity of the code. We stress that, as q is contant, maximum number of bits that can be reconstructed from a query is also constant. Hence, for any other decoding algorithm, this argument will cover constant fraction of message-bit pairs (instead of every pair in linear case), which is sufficient for our arguments.

Clearly the communication complexity from the sender side is q. So if $q = o(\kappa/\log \kappa)$ (specifically when q is a constant) this contradicts Theorem 2.7.

Hence for constant q, with high probability, for all i, j there exist a probabilistic polynomial time algorithm $A_{i,j}$ such that

$$|Pr[w \leftarrow Q(i) : A_{i,j}(w) = 1] - Pr[w \leftarrow Q(j) : A_{i,j}(w) = 1]| > 1/p_1(\kappa) \tag{4}$$

where $p_1(\kappa)$ is some polynomial on security parameter κ.

7 Approximation of vertex cover in q-regular hypergraphs

Lemma 7.1 *For any q-regular hypergraph the set of vertices participating in a maximal matching is a vertex cover of size at most q times the size of minimum vertex cover.*

Proof. Let S' be a minimum vertex cover of the hypergraph, G. Consider a maximal matching M of G. Let S be the set of vertices participating in any matched edge. Now S is a vertex cover of G since if there exists any edge with no endpoint in S then M is not maximal. Also since $|S'|$ is at least $|M|$ so,

$$|S'| \leq |S| = q \times |M| \leq q|S'|.$$

■

8 Symmetric Key Locally Decodable Codes

8.1 Code Scrambling

Let (E, D) be any error correcting code. Let $E^*(x) = \pi(E(x)) + r$ where π is a random permutation of indices and r is a random binary string. Let D^* be the corresponding decoding algorithm. Lipton proved the following the result in [9]

Lemma 8.1 *If (E, D) is an error correcting code such that D can retrieve the message with probability p from a random error then (E^*, D^*) is an error correcting code such that D^* can retrieve the message with probability p from the error introduced an computationally bounded adversary.*

8.2 Proof of Theorem 1.4

In this section we consider Locally Decodable Codes in the symmetric key model. Hence the sender and the receiver shares a secret key SK, unknown to the computationally bounded adversary (see Definition 2.6). Our construction is very similar to Lipton's code scrambling approach and approach used in [11]. But unlike previous constructions, our decoding algorithm $\mathcal{D}(i, ., SK)$ is randomized. This allows $\mathcal{D}$ to work by probing only constant number of codeword bits.

Let ECCC be any error correcting code. Let (C, D) be the encoding and decoding algorithm of ECC. Let ρ be the rate of the code and D can correct up to γ fraction of errors. Using this error correcting code ECCC and Hadamard code we would construct a (q, δ, ϵ)-LDC in the symmetric key model. The secret key of the sender and the receiver will be a random permutation π on $[nl]$ and a random string r. For every message bit i, the encoding algorithm computes $\log^2 \kappa$ query pairs of decoding algorithm of the Hadamard code where each query pair is of the form $(a.x, x.(a + e_i))$ where x is the message, $a \in \{0, 1\}^n$ is decided by the encoder and e_i is the unit vector at direction i. As each bit of Hadamard code can be computed independently, the encoding algorithm needs to compute $\mathcal{O}(\log^2 \kappa)$ inner products. Next these query pairs for all the message bits are concatenated and the random permutation π is applied. Now the encoder XORs the random string r. Finally the error correcting code ECC is applied on each bit pairs.

The formal definition of our LDC is given in Section 8.3

8.3 Constant query LDC in the Symmetric Model

Key Generation $\mathcal{G}(1^\kappa)$

- Let $n' = n \log^2 \kappa$. $SK = (\pi, r)$ where π is a random permutation on $[nl]$ and $r \in \{0, 1\}^{n'}$ is a uniform random binary string.

Encoder $\mathcal{E}(x, (\pi, r))$

1. Let $|x| = n$; Set $l = \log^2 \kappa$.
2. for $i = 1$ to n
   (a) Pick l random elements $a_{i,1}, a_{i,2} \cdots, a_{i,l} \in \{0, 1\}^n$.
   (b) Compute the pair $w_{i,j} = (a_{i,j} \bullet x, (a_{i,j} + e_i) \bullet x)$ for all $j = 1, 2 \cdots, l$ where $\bullet$ denotes inner product.
3. Concatenate the pairs and construct $z = w_{1,1}||w_{1,2}|| \cdots ||w_{1,l}|| \cdots ||w_{n,2}||w_{n,2}|| \cdots ||w_{n,l}$.

4. Apply the random permutation π on the string of pairs and compute $z' = z_1||z_2||\cdots||z_{nl}$ where $w_{i,j} = z_{\pi((i-1)l+j)}$.

5. $y' = z' + r$. Let $y' = y_1||y_2||\cdots||y_{ln}$ where each y_i is a pair of bits.

6. Apply C on each y_is and send $y = C(y_1)||C(y_2)||\cdots||C(y_{ln})$.

In the above algorithm each $w_{i,j}$ (computed in Step 2b) is a possible query pair of Hadamard code to decode x_i. The use of ECC ensures that adversary has to concentrate on destroying the query pairs rather than individual bits. It is easy to check that $m = 2\rho n \log^2 \kappa$. where ρ is a constant depending on ECC.

Now to the decode message bit i, the decoding algorithm chooses one j uniformly at random from $\{1, 2, ..., \log^2 \kappa\}$. Then it applies the permutation $\pi((i-1)l + j)$ to get the position of the corresponding query pair (encoded by C). The decoding algorithm reads the corresponding bits and gets the pair of bits using D. Now applying XOR on the pair and the corresponding bits of r, adversary retrieves message bit x_i.

Decoder $\mathcal{D}(i, ., (\pi, r))$

1. Let $\overline{y}$ denotes the corrupted codeword.

2. Pick a random j from $\{1, 2, ..., \log^2 \kappa\}$.

3. Find the index of j^{th} query pair in the corrupted codeword by $k = \pi((i-1)l + j)$.

4. Read $y_{2\rho(k-1)+1}, \cdots, y_{2\rho k}$.

5. Apply $D(y_{2\rho(k-1)+1}, \cdots, y_{2\rho k})$ to obtain the pair of bits (c_1, c_2).

6. XOR the corresponding bits of r by $z_1 = c_1 + r_{2k-1}$ and $z_2 = c_2 + r_{2k}$

7. Output $b = z_1 + z_2$.

Note that the query complexity for the decoder is just 2ρ which is a constant depending on the error correcting code ECC. The following lemma proves the theorem.

Lemma 8.2 *There exist constants δ, ϵ such that $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ is a (q, δ, ϵ) private key locally decodable code.*

Proof. [of Lemma 8.2] It is clear that $\mathcal{D}$ reads $q = 2\rho$ bits of the codeword in every invocation. As ρ is constant, so is q. The decoder reads each bit pair y_i as a block of two bits. Now the application of ECC ensures that to destroy a block y_i, the adversary has to destroy at least γ fraction bits of the codeword corresponding to the block. As our adversary can corrupt at most δ fraction of the total codeword bits, the channel can corrupt at most $\beta = \delta/\gamma$ fraction of y_is in y'.

Because of Lemma 8.1 it is enough to prove that for $z = w_{1,1}||w_{2,1}||\cdots||w_{1,n}||w_{2,n}||\cdots||w_{l,n}$ and for every $i \in [n]$

$$\Pr_e[\Pr[\mathcal{D}(i, z + e, SK) = x_i] > \frac{1}{2} + \epsilon] > 1 - \nu(\kappa).$$

where the first probability is over uniform distribution of e (hamming weight of $e \leq \beta nl$), the second probability is over internal coin tosses of $\mathcal{D}$ and $\nu(\kappa)$ is negligible in security parameter κ. Suppose p is the probability over e such that for some bit i,

$$\Pr[\mathcal{D}(i, z + e, SK) = x_i] \leq \frac{1}{2} + \epsilon$$

In other terms, p is the probability that e has 1 in at least $\frac{1}{2} - \epsilon$ fraction of the query blocks corresponding to a message bit i. As e is uniformly distributed,

$$p = \frac{\binom{\ell}{\lambda}\binom{m_1-\lambda}{\beta m_1-\lambda}}{\binom{m_1}{\beta m_1}}$$

where $\ell = \log^2 \kappa$, $\lambda = (\frac{1}{2} - \epsilon)\ell$, $m_1 = \frac{m}{2} = n\ell$. If we set $\epsilon = \frac{1}{4}$ then using the method of Ostrovsky *et al* [11], one can prove $p \leq 2^{-\ell} \leq \nu(\kappa)$. Hence the lemma follows. ■