Amortized Locally Decodable Codes

Jeremiah Blocki and Justin Zhang
Department of Computer Science
Purdue University
West Lafayette, IN
Emails: {jblocki,zhan3554}@purdue.edu

***Abstract*—Locally Decodable Codes (LDCs) are error correcting codes that admit efficient decoding of individual message symbols without decoding the entire message. Unfortunately, known LDC constructions offer a sub-optimal trade-off between rate, error tolerance and locality, the number of queries that the decoder must make to the received codeword $\tilde{y}$ to recovered a particular symbol from the original message x, even in relaxed settings where the encoder/decoder share randomness or where the channel is resource bounded. We initiate the study of Amortized Locally Decodable Codes where the local decoder wants to recover multiple symbols of the original message x and the total number of queries to the received codeword $\tilde{y}$ can be amortized by the total number of message symbols recovered. We demonstrate that amortization allows us to overcome prior barriers and impossibility results. We first demonstrate that the Hadamard code achieves amortized locality below 2 — a result that is known to be impossible without amortization. Second, we study amortized locally decodable codes in cryptographic settings where the sender and receiver share a secret key or where the channel is resource-bounded and where the decoder wants to recover a consecutive subset of message symbols $[L, R]$. In these settings we show that it is possible to achieve a trifecta: constant rate, error tolerance and constant amortized locality.**

This link will no longer be manually updated. For the most updated version, see arxiv: https://arxiv.org/abs/2502.10538

I. Introduction

Locally Decodable Codes (LDCs) are error correcting codes that admit fast single-symbol decodability after making a small number of queries to the received (possibly corrupted) codeword $\tilde{\boldsymbol{y}}$. In particular, an (n, k)-code over an alphabet Σ is an $(\ell, \delta, \varepsilon)$-LDC if there exists a pair of sender/receiver algorithms $\mathsf{Enc} : \Sigma^k \to \Sigma^n$ encoding messages of length k to codewords of length n, and $\mathsf{Dec}^{\tilde{\boldsymbol{y}}} : [k] \to \Sigma$ decoding any requested single message index $i \in [k]$ where $[k] := \{1, 2, \ldots, k\}$. We require that for all messages $\boldsymbol{x}$ and received word $\tilde{\boldsymbol{y}}$, the decoder makes at most ℓ queries to $\tilde{\boldsymbol{y}}$ and if the hamming distance $d(\mathsf{Enc}(\boldsymbol{x}), \tilde{\boldsymbol{y}}) \leq \delta n$ i.e the error caused by an adversarial channel is at most δn, we require that the decoder is correct with probability at least $1 - \varepsilon$ i.e $\Pr[\mathsf{Dec}^{\tilde{\boldsymbol{y}}}(i) = \boldsymbol{x}_i] \geq 1 - \varepsilon$. The main parameters of interest in LDCs are the *rate* $R := k/n$, *locality* ℓ, and *error-rate tolerance* δ. For instance, LDC s can be used to store files, where we want the rate to be small to limit storage overhead, the error-tolerance to be high for fault tolerance, and locality to be low for ultra-efficient recovery of any portion of the file.

The trade-offs between rate, locality, and error tolerance within LDCs for (worst-case) classical errors have been extensively studied yet achievable parameters remain sub-optimal and undesirable. Ideally, we would like an LDC which achieves constant rate, constant locality, and constant error-rate tolerance simultaneously. However, any LDC with constant locality $\ell \geq 2$ or constant error tolerance $\delta > 0$ must have at least non-linear rate [1], where the best known constructions (e.g Hadamard and matching vector codes) have super-polynomial rate [2]. In particular, for locality $\ell = 2$, we have constructions, but at the same time, any construction must have exponential rate [2]–[5]. Katz and Trevisan show further that there do not exist LDC for $\ell < 2$ even when the rate is allowed to be exponential [1]. It is easy to verify that their result further extends to settings where the error-rate is $\delta = o(1)$ e.g $\delta = O(1/n^{.99})$. In other words, a local decoder must read *at least twice* as much information as requested in the setting of worst-case errors.

Various relaxations have been introduced to deal with these undesirable trade-offs for classical LDCs. For example, Ben-Sasson et al. introduced the notion of a relaxed LDCs [6], allowing the decoder to reject (output $\perp$) instead of outputting a codeword symbol whenever it detects error. This was further expanded by Gur et al. to study locally *correctable* codes where the decoder returns symbols of the codeword instead of the message [7]. Another line of work studies LDCs that allow a sender and receiver to share a secret key that is unknown to the computationally-bound channel [8]–[10]. Yet another line of work considers LDCs in settings where the channel is resource bounded (e.g it cannot evaluate circuits beyond a particular size/depth) [11].

However, even with these relaxations, the achievable trade-offs are still sub-optimal. For relaxed locally decodable/correctable codes respectively, Ben-Sasson et al. and Gur et al. are able to achieve LDC/LCC constructions with constant locality and constant error-tolerance codes, but with sub-optimal codeword length $n = O\left(k^{1+O(1)/\sqrt{\ell}}\right)$ [6], [12]. Gur et al. also prove that any relaxed LDC must have codeword length $n = \Omega\left(k^{1+c}\right)$, where $c = 1/O\left(\ell^2\right)$ [13], ruling out any possibility of having constant rate, constant locality, and constant error-tolerance. In fact, any relaxed LDC with constant error tolerance, perfect completeness, and locality $\ell = 2$ must have exponential rate [14]. Moreover, constructions for relaxed LDCs with constant error-rate tolerance, constant rate, and locality $\ell = O(\mathsf{polylog}(k))$ are unknown. While these parameters *are* possible in the shared-cryptographic-key [8]–[10] and resource-bounded-channel settings [11], [15], there are no constant rate, constant error-rate tolerance, constant locality constructions.

Traditionally, in LDC literature, the decoder is tasked with recovering the symbol or bit at a single index. However, most practical applications desire the recovery of a much larger portion of the message. For example, the local decoder may want to recover the symbol $\boldsymbol{x}_i$ for *every* index $i \in Q$ for some set $Q \subseteq [k]$. The natural and naive way to accomplish this task would be to run a local decoder $|Q|$ times separately for each $i \in Q$, but the total query complexity will be $\ell|Q|$. In this paper, we ask the following natural question: is it possible to improve the total query complexity beyond that of the naive solution by designing a decoder that attempts to decode all requested symbols in one run, *amortizing* the number of queries?

Our Contributions: We initiate the study of *amortized locally decodable codes* which seek to reduce query complexity by amortizing the local decoding process. Given a set $Q \subseteq [k]$, the local decoding algorithm $\mathsf{Dec}^{\tilde{\boldsymbol{Y}}}(Q)$ should output $\{\boldsymbol{x}_i\}_{i \in Q}$ where the amortized query complexity α is given by the total number of queries made by the decoder ℓ divided by the total number of message symbols recovered i.e., $\alpha = \ell/|Q|$.

We first show that the Hadamard code [2] can achieve *amortized* locality $\alpha < 2$. In fact, if the the error-rate is $\delta = o(1)$ then the amortized locality approaches 1. This stands in stark contrast to the impossibility results of Katz and Trevisan who proved that, without amortization, any LDC must have $\ell \geq 2$ even if $\delta = o(1)$.

Second, we study amortized locally decodable codes in cryptographic settings where the sender and receiver share a secret key. We show that when the decoder wants to recover a *consecutive* subset of bits $[L, R] \subset [k]$ that it is possible to achieve *constant rate, constant error tolerance and constant amortized locality*. To the best of our knowledge this is the first construction which achieves all three goals simultaneously, even in the setting where the sender/receiver share a secret key.

Finally, we can apply the framework of [11], [15], [16] to remove the assumption that the sender and receiver share a secret key as long as the channel is resource bounded and is unable to solve cryptographic puzzles. As Blocki et al. [11] argued, resource bounded channels can plausibly capture any error pattern that arises naturally, that is, real-world channels are resource bounded. For example, suppose that A denotes a randomized algorithm that models the error pattern of our channel. If the channel has small latency then we can reasonably assume that the algorithm A must have low-depth — there may be many computational steps if the algorithm A is parallel, but the depth of the computation is bounded. This means that a low-latency channel would be incapable of solving time-lock puzzle [17] — cryptographic puzzles that are solvable in t sequential computation steps, but cannot be solved in $o(t)$ time by any parallel algorithm running in polynomial time. One can also design cryptographic puzzles that are space-hard meaning that they cannot be solved by any probabilistic polynomial time algorithm using space $o(s)$, but can be solved easily using space s. Additionally, other categories of resource-bounded cryptographic puzzles exist, such as memory-hard [15], space-hard puzzles, which impose constraints on time-space complexity and space complexity respectively. Ameri et al. [15] showed how to use cryptographic puzzles and secret key LDCs to construct resource-bounded LDCs with constant rate, constant error tolerance, but their locality is $O(\text{polylog}(k))$. We demonstrate that this construction achieves amortized locality $O(1)$ if we use our amortizable secret key LDC.

II. Amortized Locality

We provide the first formalization of amortized locally decodable codes. We say two strings g and h of the same length are δ-close if g has hamming distance at most $\delta|g|$ from h.

Definition 1: A (n,k)-code $\mathcal{C}$ is a $(\alpha,\kappa,\delta,\epsilon)$-amortizeable LDC (aLDC) if there exists an algorithm Dec such that for every $x \in \Sigma^k$, $\tilde{y} \in \Sigma^n$ such that $\tilde{\boldsymbol{y}}$ is δ-close to $\mathcal{C}(x)$, and every subset $Q \subseteq [k]$ with $|Q| \geq \kappa$ we have

$$\Pr[\mathsf{Dec}^{\tilde{y}}(Q) = \{\boldsymbol{x}_i : i \in Q\}] \geq 1 - \varepsilon$$

and $\mathsf{Dec}^{\tilde{y}}$ makes at most $\alpha|Q|$ queries to $\tilde{y}$.

An $(\alpha,\kappa,\delta,\varepsilon)$-aLDC permits that the decoder make up to $|Q|\alpha$ total queries when attempting to decode the target symbols in the set Q. The amortized number of queries per symbol is just α, but because the decoder may make up to $|Q|\alpha$ queries in total, it may be possible to circumvent classical barriers and impossibility results.

As a first motivation for aLDCs we first consider an impossibility result of Katz and Trevisan [1] who proved that any LDC must have locality $\ell \geq 2$. In particular, they proved that for any $(1,\delta,\epsilon)$-LDC we have $k \leq \frac{\log|\Sigma|}{\delta(1-H(1/2+\varepsilon))}$ where $H(\cdot)$ is the entropy function in base $|\Sigma|$. Even if we set the error-tolerance to $\delta = 1/\sqrt{k}$, so that $\delta = o(1)$, we still have the constraint that $\sqrt{k} \leq \frac{\log|\Sigma|}{1-H(1/2+\epsilon)}$. Thus, it is impossible to construct a $(1,\delta,\epsilon)$-LDC which supports arbitrarily long messages in Σ^k. It follows that any LDC construction that supports long messages must have locality $\ell \geq 2$. We show that it is possible to break this barrier by amortizing the decoding costs across multiple queries and achieve amortized locality $1 + O(\delta/\epsilon)$. Note that if ϵ is a constant and $\delta = o(1)$, the amortized locality is $1 + o(1)$, that is, the amortized locality approaches 1. In fact, we show that the Hadamard code already achieves these properties [18] — see Theorem 2.

The Hadamard code encodes binary messages $\boldsymbol{x} \in \{0,1\}^k$ of length k to binary codewords $\boldsymbol{y}$ of length 2^k, where each codeword bit corresponds to a XOR of message bit subsets, $\boldsymbol{y}_S := \oplus_{i\in S}\boldsymbol{x}_i, \forall S \subseteq [k]$. More precisely, $\mathsf{Enc}(\boldsymbol{x}) = \langle \boldsymbol{y}_S \rangle_{S\subseteq[k]} \in \{0,1\}^{2^k}$ where $\boldsymbol{y}_S := \oplus_{i\in S}\boldsymbol{x}_i$. We show that by extending a simple idea from a traditional single bit local decoder, we can have amortize beyond what is possible for traditional locality.

A simple decoder achieving 2-locality for Hadamard codes decodes any message bit index i by selecting a random subset $S \subseteq [k]$, computing the subset $S_i := S\Delta\{i\}$ (Δ denotes symmetric-difference), querying the received codeword $\tilde{\boldsymbol{y}}$ at the indices corresponding to S and S_i to obtaining $\tilde{\boldsymbol{y}}_S$ and $\tilde{\boldsymbol{y}}_{S_i}$ and then outputting $\tilde{\boldsymbol{y}}_S \oplus \tilde{\boldsymbol{y}}_{S_i}$. If the error-rate is set to δ, then by a union bound, with probability at least $1 - 2\delta$ we have $\tilde{\boldsymbol{y}}_S = \boldsymbol{y}_S$ and $\tilde{\boldsymbol{y}}_{S_i} = \boldsymbol{y}_{S_i}$ i.e., both queried bits are correct. If both queried bits are correct, then the decoder will succeed as $\tilde{\boldsymbol{y}}_S \oplus \tilde{\boldsymbol{y}}_{S_i} = \boldsymbol{y}_S \oplus \boldsymbol{y}_{S_i} = \boldsymbol{x}_i$.

If we want to recover multiple message bits, then we can instead pick a random set S and then query to obtain $\tilde{\boldsymbol{y}}_S$ and $\tilde{\boldsymbol{y}}_{S_j}$ for all $j \in Q$. The total number of queries will be $|Q|+1$ so the amortized locality is just $1+1/|Q|$. By union bounds we will have $\tilde{\boldsymbol{y}}_S = \boldsymbol{y}_S$ and $\tilde{\boldsymbol{y}}_{S_j} = \boldsymbol{y}_{S_j}$ for all $j \in Q$ with probability at least $1 - \delta(|Q|+1)$. These observations lead to Theorem 2.

Theorem 2: For any $k,\delta,\kappa > 0$, the Hadamard code is a $(2^k-1,k)$-code that is also a $\left(\frac{\kappa+1}{\kappa},\kappa,\delta,\varepsilon\right)$-aLDC, where $\varepsilon \geq (\kappa+1)\delta$.

For example, if $\varepsilon \leq \frac{1}{3}$ and $\delta \leq \frac{1}{9}$ then we have $\left(\frac{3}{2},2,\delta,\varepsilon\right)$-aLDC. In fact, if $\delta = o(1)$, then we have an aLDC with amortized locality $\alpha \to 1$ as we can set $\kappa + 1 = \epsilon/\delta$ so that $\kappa \to \infty$. This is in stark contrast to the result of Katz and Trevisan, which state that (without amortization) no LDCwith locality $\ell < 2$ exists even when $\delta = o(1)$.

III. Private Locally Decodable Codes

In the previous section we saw how amortization allowed us to push past the locality $\ell = 2$ barrier and achieve amortized locality $\alpha < 2$ with constant error-tolerance, The primary downside to the Hadamard construction is that the rate R is exponential. Ideally, we want a construction with constant rate, constant error tolerance and constant amortized locality. It remains an open question whether or not this goal is achievable. As an initial step we show that the goal is achievable in relaxed settings where the sender and receiver share randomness or where the channel is computationally bounded. In this section we will also make the natural assumption that the decoder wants to recover a consecutive portion of the original message i.e., $Q = [L, R] = \{L, L+1, \ldots, R\} \subseteq [k]$.

In settings where the sender and receiver share randomness (e.g., cryptographic keys) or where the channel is resource-bounded we can slightly relax the correctness condition for an aLDC. Recall that we previously required the decoder $\mathsf{Dec}^{\tilde{\boldsymbol{y}}}$ succeed with probability at least $1-\epsilon$ for any corrupted codeword $\tilde{\boldsymbol{y}}$ that is sufficiently close to the original codeword $\boldsymbol{y}$. In relaxed versions of the definition, it is acceptable if there exists corrupted codewords $\tilde{\boldsymbol{y}}$ that fool the decoder and are close to the original codeword, as long as it is computationally infeasible for an adversarial, but resource-bounded channel to find such a corruption with high probability. This motivates definition 3.

Let $\leftarrow$ denote a probabilistic assignment where $\stackrel{\$}{\leftarrow}$ emphasizes a uniformly random assignment.

Definition 3: Let λ be the security parameter. A triple of probabilistic polynomial time algorithms $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is a private $(\alpha, \kappa, \delta, \epsilon, q)$-amortizeable LDC ($\mathsf{paLDC}$) if

- for all keys $\mathsf{sk} \in \mathrm{Range}(\mathsf{Gen}(1^\lambda))$ the pair $(\mathsf{Enc}_{\mathsf{sk}}, \mathsf{Dec}_{\mathsf{sk}})$ is an (n,k)-code, and
- for all probabilistic polynomial time algorithms $\mathcal{A}$ there is a negligible function μ such that

$$\Pr[\texttt{paLDC-Sec-Game}(\mathcal{A}, \lambda, \delta, \kappa, q) = 1] \leq \mu(\lambda),$$

where the probability is taken over the randomness of $\mathcal{A}$, Gen, and `paLDC-Sec-Game`. The experiment `paLDC-Sec-Game` is defined as follows:

paLDC-Sec-Game$(\mathcal{A}, \lambda, \delta, \kappa, q)$

The challenger generates secret key $\mathsf{sk} \leftarrow \mathsf{Gen}(1^\lambda)$. For q rounds, on iteration h, the challenger and adversary $\mathcal{A}$ interact as follows:

1) The adversary $\mathcal{A}$ chooses a message $x^{(h)} \in \{0,1\}^k$ and sends it to the challenger.
2) The challenger sends $y^{(h)} \leftarrow \mathsf{Enc}(\mathsf{sk}, x^{(h)})$ to the adversary.
3) The adversary outputs $\tilde{y}^{(h)} \in \{0,1\}^n$ with hamming distance at most δn from $y^{(h)}$.
4) If there exists $L^{(h)}, R^{(h)} \in [k]$ such that $R^{(h)} - L^{(h)} + 1 \geq \kappa$ and

$$\Pr\left[\mathsf{Dec}_{\mathsf{sk}}^{\tilde{y}^{(h)}}(L^{(h)}, R^{(h)}) \neq x_L^{(h)} \cdots x_R^{(h)}\right] > \varepsilon(\lambda)$$

such that $\mathsf{Dec}_{\mathsf{sk}}^{\tilde{y}^{(h)}}(.)$ makes at most $(R - L + 1)\alpha$ queries to $\tilde{y}$, then this experiment outputs 1.

If the experiment did not output 1 on any iteration h, then output 0.

A. One-Time paLDC

Our first paLDC construction will be based on the private-key construction of Ostrovsky et al. [19]. The secret key in our scheme will be a random permutation π and a one-time pad $\boldsymbol{R}$. To encode the message $\boldsymbol{x}$, first split it into B equal-sized blocks of size $a = \omega(\log \lambda)$, where $\boldsymbol{x} = \boldsymbol{w}_1 \circ \ldots \boldsymbol{w}_B$ ($\circ$ is the concatenation function). Encode each block $\boldsymbol{w}_j$ as $\boldsymbol{w}'_j = \mathcal{C}(\boldsymbol{w}_j)$, where $\mathcal{C}$ is a code with a constant rate R and constant error-tolerance δ. Form the encoded message as $\boldsymbol{y}' = \boldsymbol{w}'_1 \circ \cdots \circ \boldsymbol{w}'_B$, where $|\boldsymbol{y}|/|\boldsymbol{x}| = 1/R$. Note that we cannot just output $\boldsymbol{y}'$ without assuming sub-constant error-tolerance because otherwise, an adversarial channel can just choose to corrupt an entire block $\boldsymbol{w}'_j$. To remedy this, we apply our secret random permutation π and the one-time pad $\boldsymbol{R}$ to output $\boldsymbol{y} = \pi(\boldsymbol{y}' \oplus \boldsymbol{R})$. Since the channel is computationally bounded, the errors it causes is effectively random. If the overall error tolerance is a constant dependent on δ, then each block will have at most δn errors with high probability (see [19] for details).

Thus, our local decoder will simply recover its requested message symbols by recovering the corresponding message blocks. That is, if block $\boldsymbol{w}_j$

is requested, then we undo the permutation and one-time pad to obtain the encoded block $\boldsymbol{w}'_j$ and subsequently decode it. More specifically, for each index j_r of $\boldsymbol{w}'_j$ in $\boldsymbol{y}'$, we obtain the corresponding index in $\boldsymbol{y}$ as $\pi(j_r)$. In summary, this code achieves constant rate, constant error-tolerance, and constant amortized locality with parameters summarized in the following theorem.

Theorem 4: Suppose code $\mathcal{C}$ has constant rate R and constant error-tolerance. Then, the construction above is a $(2/R, \omega(\log \lambda), O(1), O(1), 1)$-$\mathsf{paLDC}$.

The primary limitation to the above construction is that the security is only guaranteed after the encoder sends a *single* ($q = 1$) message to the decoder. If the encoder has multiple messages to send, then they would need to use a separate permutation π^i and a one-time pad $\boldsymbol{R}^i$ in every round $i \leq q$. Generating and secretly sharing randomness for each message is costly and undesirable; instead, we propose an alternate where the secret key may be used polynomial many times. The full analysis can be found in appendix B.

B. *Multi-Round* paLDC

We present a polynomial round ($q = \mathsf{poly}(\lambda)$) paLDC with constant rate, constant error rate tolerance, and constant amortized locality that matches the one-time construction without requiring multiple secret keys. Our primary technical ingredient is a special type of code that we call a *robust secret encryption (*RSE*)*. Intuitively, we want a code with the property that any computationally bounded adversary who does not have the secret key for the scheme cannot distinguish the encoding of a random message from a random string. This allows us to embed fresh randomness in such a way that the randomness is effectively hidden from an attacker who does not have the secret key.

Formally, the definition of a RSE is given in definition 5. Intuitively, the RSE game captures the property that a PPT attacker cannot distinguish the encoding of a random message from a truly random string even if the attacker is given many samples. Recent work [20] yields an efficient construction (constant rate/error tolerance) of RSE from the Learning Parity with Noise (LPN) assumption — a standard widely accepted assumption in the field of cryptography. We also provide a construction based on the Gopppa code McEliece Cryptosystem, which can be found in appendix A.

Definition 5: A (n, k, δ)-Robust Secret Encryption (RSE) is a tuple of probabilistic polynomial time algorithms $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ such that:

- For all keys $\mathsf{sk} \in Range(\mathsf{Gen}(1^\lambda))$, $(\mathsf{Enc}_{\mathsf{sk}}, \mathsf{Dec}_{\mathsf{sk}})$ is a (n, k) code that can tolerate δn errors.
- For any probabilistic polynomial time algorithm $\mathcal{A}$ playing the $\texttt{RSE-Game}$, $q \in \mathsf{poly}(\lambda)$, there exists a negligible function ε such that

$$\left|\Pr[\texttt{RSE-Game}(\mathcal{A}, \lambda, q) = 1] - \frac{1}{2}\right| < \varepsilon(\lambda),$$

where the $\texttt{RSE-Game}$ is defined as,

$\texttt{RSE-Game}$ $(\mathcal{A}, \lambda, q)$

The challenger generates $\mathsf{sk} \leftarrow \mathsf{Gen}(1^\lambda)$ and $b \xleftarrow{\$} \{0, 1\}$ then sends $\mathcal{A}$ a sequence $\{R^i\}_{i \in [q]}$, where each R^i is (identically and independently) generated as follows:

- if $b = 0$, $R^i \leftarrow \mathsf{Enc}_{\mathsf{sk}}(r^i)$ where $r^i \xleftarrow{\$} \mathbb{F}^k$,
- otherwise if $b = 1$, $R^i \xleftarrow{\$} \mathbb{F}^n$.

$\mathcal{A}$ outputs bit $b' \in \{0, 1\}$, and if $b = b'$, the output of this experiment is 1. Otherwise, the output is 0.

Lastly, we will need a common cryptographic tool known as the pseudorandom function (prf). Informally, a prf is a deterministic function f that when instantiated with a secret key $\boldsymbol{k}$, is indistinguishable from a random function to a computationally-bound adversary. We use the prf to essentially generate a new one-time pad for each message sent, allowing us to invoke the indistinguishability of the RSE.

Construction 6: Let $\mathsf{RSE}.(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be an (A, a, δ)-RSE with rate R_{RSE}, and let $f : \{0,1\}^\lambda \times \{0,1\}^{b + \log B} \to \{0,1\}^a$ be a prf f, where $a = \mathsf{poly}(b)$.

$\mathsf{Gen}(1^\lambda)$

Output $\mathsf{sk} \leftarrow (\pi, \boldsymbol{k}, \mathsf{sk}')$ where $\pi \xleftarrow{\$} S_{BA}$, $\boldsymbol{k} \xleftarrow{\$} \{0,1\}^\lambda$, and $\mathsf{sk}' \leftarrow \mathsf{RSE}.\mathsf{Gen}(1^\lambda)$.

$\mathsf{Enc}_{\mathsf{sk}}(\boldsymbol{x})$

Parse $(\pi, \boldsymbol{k}, \mathsf{sk}') \leftarrow \mathsf{sk}$.

For each $i = 1, \ldots, B$:

1) Let $\boldsymbol{w}_i = x_{ia+1} \cdots x_{(i+1)a}$.
2) Generate $\boldsymbol{r}_i \xleftarrow{\$} \{0,1\}^b$.
3) Let $\boldsymbol{z}_i = f_{\boldsymbol{k}}(i \circ \boldsymbol{r}_i)$.
4) Let $\boldsymbol{w}'_i = \mathsf{RSE.Enc}_{\mathsf{sk}'}((\boldsymbol{w}_i \oplus \boldsymbol{z}_i) \circ \boldsymbol{r}_i)$.

Let $\boldsymbol{y}' = \boldsymbol{w}'_1 \circ \ldots \boldsymbol{w}'_B$ and output $\boldsymbol{y} \leftarrow \pi(\boldsymbol{y}')$.

$\mathsf{Dec}^{\tilde{\boldsymbol{y}}}_{\mathsf{sk}}(L, R)$

Parse $(\pi, \boldsymbol{k}, \mathsf{sk}') \leftarrow \mathsf{sk}$.

Suppose $\boldsymbol{x}[L, R]$ lies in $\boldsymbol{w}_{i+1} \circ \cdots \circ \boldsymbol{w}_{i+\ell}$ for some $i \in [B - \ell]$. For each $j = i+1, \ldots, i+\ell$:

1) Let $j_1, \ldots, j_A$ be the indices of $\boldsymbol{w}'_j$ in $\boldsymbol{y}'$.
2) Let $\boldsymbol{w}'_j = \widetilde{\boldsymbol{y}}_{\pi(j_1)} \circ \cdots \circ \widetilde{\boldsymbol{y}}_{\pi(j_A)}$ be obtained by querying $\widetilde{y}$ at those A indices.
3) Compute $(\boldsymbol{d}_{j,1} \circ \boldsymbol{d}_{j,2}) \leftarrow \mathsf{RSE.Dec}(\boldsymbol{w}'_j)$
4) Compute $\boldsymbol{w}_j = \boldsymbol{d}_{j,1} \oplus f_{\boldsymbol{k}}(j \circ \boldsymbol{d}_{j,2})$

From $\boldsymbol{w}_{i+1}, \ldots, \boldsymbol{w}_{i+\ell}$, output bits corresponding to $\boldsymbol{x}[L, R]$.

Theorem 7: Suppose $(\mathsf{Gen}_{\mathsf{RSE}}, \mathsf{Enc}_{\mathsf{RSE}}, \mathsf{Dec}_{\mathsf{RSE}})$ is a $(A, a, \delta_{\mathsf{RSE}})$-RSE with rate $R_{\mathsf{RSE}} = a/A$. Then, construction 6 is a $\left(\frac{2+o(1)}{R_{\mathsf{RSE}}}, a, \delta, \varepsilon\right)$-paLDC, where when $\delta < \delta_{\mathsf{RSE}}$, ε is negligible.

Proof: For any $L, R \in [k]$ such that $R - L + 1 \geq \kappa$, suppose $x[L, R]$ lie in blocks $w_{s+1}, \ldots, w_{s_{s+\ell}}$. Then, $\ell \leq \lfloor \frac{R-L+1}{a} \rfloor + 1$. To recover each of these w_j blocks from $\widetilde{y}$, the decoder accesses the corresponding encrypted block w'_j. Thus, the decoder accesses $\ell A = \ell \times \frac{(a+b)}{R_{\mathsf{RSE}}}$ bits in total and we have that

$$\begin{aligned}
\alpha &\leq \frac{\ell(a+b)/(R-L+1)}{R_{\mathsf{RSE}}} \\
&\leq \frac{\left(\lfloor \frac{R-L+1}{a} \rfloor + 1\right)(a+b)/(R-L+1)}{R_{\mathsf{RSE}}} \\
&\leq \frac{\left(\frac{1}{a} + \frac{1}{R-L+1}\right)(a+b)}{R_{\mathsf{RSE}}} \\
&\leq \frac{2 + 2\frac{b}{a}}{R_{\mathsf{RSE}}}, \qquad (R - L + 1 \geq a)
\end{aligned}$$

where the term $2b/a \in o(1)$.

We now upper bound $\Pr[\mathtt{paLDC\text{-}Sec\text{-}Game}(\mathcal{A}, \lambda, \delta, \kappa, q) = 1]$ by upper bounding the the probability of the event $\mathtt{BAD} = \bigcup_{i \leq q, j \leq B} \mathtt{BAD}^i_j$ where $\mathtt{BAD}^i_j$ is the event that in round i block $\boldsymbol{w}'_j$ has more than $\delta_{\mathsf{RSE}} A$ errors. As long as the event $\mathtt{BAD}$ does not occur it is guaranteed that the local decoder will be successful in all rounds.

We proceed by defining a series of modified games (or Hybrids), where we argue that the incorrect decoding probability difference from the original game only differs negligibly.

We define the series of Hybrids H_0 to H_4 as follows: Denote round by superscript notation. Then,

1) H_0: The game is played as-is.
2) H_1: Same as H_0, except that we update line 3 of the encoding algorithm to $z_i \xleftarrow{\$} \{0,1\}^A$ i.e., we replace each pseudorandom string $f_{\boldsymbol{k}}(j \circ \boldsymbol{r}_i)$ with a truly random string $\boldsymbol{R}_j \xleftarrow{\$} \{0,1\}^A$ for each block $j \in [B]$.
3) H_2: Same as H_1, except that we update line 4 of the encoding algorithm to $\boldsymbol{w}'_i = \mathsf{RSE.Enc}_{\mathsf{sk}'}(\boldsymbol{R}_j)$ where $\boldsymbol{R}_j \xleftarrow{\$} \mathbb{F}^{a + \lg B}$ instead of $\boldsymbol{w}'_i = \mathsf{RSE.Enc}_{\mathsf{sk}'}(\boldsymbol{w}_j \oplus \boldsymbol{R}_j) \circ r^i_j$.
4) H_3: Same as H_2, but we update line 4 of the encoding algorithm to set $\boldsymbol{w}_i \xleftarrow{\$} \mathbb{F}^A$ i.e., replacing $\mathsf{RSE.Enc}_{\mathsf{sk}'}$ with a uniformly random string.
5) H_4: Same as H_3, but in each round i we sample a fresh permutation π_i and output $\boldsymbol{y} = \pi_i(y')$.

The indistinguishability of Hybrids H_0 and H_1 follow from PRF security and negligible collision probability. First, disregarding possible collisions, the prf output $\boldsymbol{z}^j_i = f_{\boldsymbol{k}}(i \circ \boldsymbol{r}^j_i)$ is computationally indistinguishable from a random $\boldsymbol{R}^i_j$. Otherwise, a distinguisher for the prf can be constructed from the distinguisher of these Hybrids. Next, there is added distinguishing probability when the same prf input is used across rounds i.e. when $j \circ \boldsymbol{r}^i_j = j \circ \boldsymbol{r}^{i'}_j$ for some $i \neq i' \in [q]$. Since the probability of a collision is at most $q^2/2^a$, Hybrids H_0 and H_1 remain computationally indistinguishable. Hybrids H_1 and H_2 are statistically indistinguishable because they are the same distribution. Hybrids H_2 and H_3 are computationally indistinguishable by the security of RSE. Recall that by the security property of RSE, a computationally bound adversary cannot distinguish between polynomial-many random words and RSE encodings with random inputs. It follows that if

Hybrid's Codewords	*Justification*
$H_0 : \{\pi(\mathsf{RSE.Enc}_{\mathsf{sk}'}((\boldsymbol{w}_j^i \oplus f_{\boldsymbol{k}}(.)) \circ \boldsymbol{r}_j^i))_{j\in[B]}\}_{i\in[q]}$	Original
$H_1 : \{\pi(\mathsf{RSE.Enc}_{\mathsf{sk}'}((\boldsymbol{w}_j^i \oplus \underline{\boldsymbol{R}_j^i}) \circ \boldsymbol{r}_j^i))_{j\in[B]}\}_{i\in[q]}$	PRF Indist.
$H_2 : \{\pi(\mathsf{RSE.Enc}_{\mathsf{sk}'}(\underline{\boldsymbol{R}_j^{'i}}))_{j\in[B]}\}_{i\in[q]}$	Same Dist.
$H_3 : \{\pi(\underline{\boldsymbol{R}_j^{''i}})_{j\in[B]}\}_{i\in[q]}$	RSE Indist.
$H_4 : \{\underline{\pi^i}(\boldsymbol{R}_j^{''i})_{j\in[B]}\}_{i\in[q]}$	Same Dist.

TABLE I
HYBRID SUMMARY FOR THEOREM 7

Hybrids H_2 and H_3 were distinguishable, this would contradict the RSE security property. Lastly, Hybrids H_3 and H_4 are statistically indistinguishable since they form the same distribution. Justification for why each Hybrid is indistinguishable with respect to decoding error is summarized in the table I.

Thus, it suffices to upper bound the probability of the event $\mathtt{BAD}$ in Hybrid H_4 where a fresh random permutation π_i is used in each round i.

Since a new permutation π^j with a uniformly random mask $\boldsymbol{R}_j^{''i}$ is used in every round j, an adversary's δAb errors are uniformly distributed in each block of size A. Thus, the number of errors for a given block j is hyper-geometric$(AB, \delta AB, A)$, and by [21], [22], we have

$$\Pr[\mathtt{BAD}_j^i] < 2^{\frac{-2(((\delta_{\mathsf{RSE}}-\delta)A)^2-1)}{A+1}}$$

which is negligible with respect to $A = a/R_{\mathsf{RSE}}$ as long $\delta_{\mathsf{RSE}} > \delta$. By applying a union bound over all B blocks and q rounds, we have that there is an error decoding any block is negligible. ■

C. aLDC*s for Resource-bounded Channels*

Lastly, we present an aLDC for resource-bounded channels with constant rate, constant amortized locality, and constant error-tolerance by applying the framework developed by Ameri et al. [15] to eliminate the requirement that the encoder and decoder have a shared secret key. The framework of Ameri et al. [15] using two building blocks: a secret key LDC and cryptographic puzzles. Intuitively, a cryptographic puzzle consists of two algorithms $\mathtt{PuzzGen}$ and $\mathtt{PuzzSolve}$. $\mathtt{PuzzGen}(s)$ is a randomized algorithm that takes as input a string s and outputs a puzzle Z whose solution is s i.e., $\mathtt{PuzzSolve}(Z) = s$. The security requirement is that for any adversary $A \in \mathcal{C}$ is a class $\mathcal{C}$ of resource bounded algorithms (e.g., bounded space, bounded computation depth, bounded computation) cannot solve the puzzle Z. In fact, we require that for any string s_0 and any resource bounded adversary $A \in \mathcal{C}$ the adversary A cannot even distinguish between (Z_0, s_0, s_1) and (Z_1, s_0, s_1) where s_i is a random string and $Z_i = \mathtt{PuzzGen}(s_i)$ is a randomly generated puzzle whose corresponding solution is s_i.

At a high level the encoding algorithm $\mathsf{Enc}(x)$ works as follows: 1) pick a random string $r \in \{0,1\}^\lambda$ and generate a cryptographic puzzle $Z = \mathtt{PuzzGen}(r)$ whose solution is r. 2) Use a constant rate error correcting code to obtain an encoding C_Z of this puzzle. 3) Use the random string r to generate the cryptographic key sk for a secret key LDC (we will use the amortizeable secret key LDC (paLDC) from Theorem 7). 4) Use the secret key LDC to encode the message and obtain $c_1 = \mathsf{Enc}_{sk}(x)$. 5) Define $c_Z^1 = C_Z$ and $C_Z^{i+1} = C_Z \circ C_Z^i$ and find the smallest value r such that C_Z^r is at least as long as c_1. Set $c_2 = C_Z^r$. 6) Output the final codeword $C = c_1 \circ c_2$.

Intuitively, if the channel is resource bounded then the channel cannot solve the puzzle Z or extract any meaningful information about the solution r or the secret key sk derived from it. In contrast, the local decoding algorithm can extract several (noisy) copies of C_Z by querying c_2 and decode these copies to extract Z (most noisy copies of C_Z in c_2 will still decode to Z). Then, the decoder, who does not have the same resource constraints as the channel, can solve the puzzle Z to obtain r and then extract the secret key sk using r. Finally, once the decoder has sk it can run the (amortizeable) secret-key local decoder on c_1 to extract the message symbols that we want.

If we instantiate this construction with a paLDC, then the amortized locality is nearly the same. The local decoder needs to make $O(\lambda\ \mathtt{poly}(1/\epsilon))$ additional queries to c_2 to ensure that we recover the correct puzzle Z with high probability e.g., at least $1-\epsilon/2$. However, these additional $O(\lambda\ \mathtt{poly}(1/\epsilon))$ queries can be amortized over the total number of symbols that are decoded. We observe that our amortization block size may be made to be much larger than the key size, adding negligible amortized locality.

Theorem 8 (Informal): Suppose the channel is resource-bounded and there exists a cryptographic puzzle. Suppose $\mathcal{C}_P$ is a $(\alpha_p, \kappa_p, \delta_p, \varepsilon_p, 1)$-$\mathsf{paLDC}$. Then, under the framework of Ameri et al. [15], we can construct a $(\alpha_p + o(1), \kappa_p, \delta, \varepsilon)$-$\mathsf{aLDC}$, where $\delta = O(\delta_p)$ and $\varepsilon = O(\varepsilon_p)$.

IV. Conclusion

We initiate the study of amortized LDCs as a tool to overcome prior barriers and impossibility results. We show that it is possible to design an amortized LDC with amortized locality $\alpha < 2$ — overcoming an impossibility result of Katz and Trevisan for regular LDCs. We also design a secret-key LDC with constant rate, constant error tolerance, and constant amortized locality. Finally, under the natural assumption that the channel is resource bounded, we can use cryptographic puzzles to eliminate the requirement that the sender/ sender and obtain LDCs with constant rate, constant error tolerance, and constant amortized locality.

References

[1] J. Katz and L. Trevisan, "On the efficiency of local decoding procedures for error-correcting codes," in *32nd Annual ACM Symposium on Theory of Computing*. Portland, OR, USA: ACM Press, May 21–23, 2000, pp. 80–86.

[2] S. Yekhanin *et al.*, "Locally decodable codes," *Foundations and Trends® in Theoretical Computer Science*, vol. 6, no. 3, pp. 139–255, 2012.

[3] I. Kerenidis and R. de Wolf, "Exponential lower bound for 2-query locally decodable codes via a quantum argument," in *35th Annual ACM Symposium on Theory of Computing*. San Diego, CA, USA: ACM Press, Jun. 9–11, 2003, pp. 106–115.

[4] O. Goldreich, H. Karloff, L. J. Schulman, and L. Trevisan, "Lower bounds for linear locally decodable codes and private information retrieval," *computational complexity*, vol. 15, pp. 263–296, 2006.

[5] A. Ben-Aroya, O. Regev, and R. de Wolf, "A hypercontractive inequality for matrix-valued functions with applications to quantum computing and LDCs," in *49th Annual Symposium on Foundations of Computer Science*. Philadelphia, PA, USA: IEEE Computer Society Press, Oct. 25–28, 2008, pp. 477–486.

[6] E. Ben-Sasson, O. Goldreich, P. Harsha, M. Sudan, and S. P. Vadhan, "Robust PCPs of proximity, shorter PCPs and applications to coding," in *36th Annual ACM Symposium on Theory of Computing*, L. Babai, Ed. Chicago, IL, USA: ACM Press, Jun. 13–16, 2004, pp. 1–10.

[7] T. Gur, G. Ramnarayan, and R. D. Rothblum, "Relaxed locally correctable codes," in *ITCS 2018: 9th Innovations in Theoretical Computer Science Conference*, A. R. Karlin, Ed., vol. 94. Cambridge, MA, USA: Leibniz International Proceedings in Informatics (LIPIcs), Jan. 11–14, 2018, pp. 27:1–27:11.

[8] R. Ostrovsky, O. Pandey, and A. Sahai, "Private locally decodable codes," Cryptology ePrint Archive, Report 2007/025, 2007. [Online]. Available: https://eprint.iacr.org/2007/025

[9] B. Hemenway and R. Ostrovsky, "Public-key locally-decodable codes," in *Advances in Cryptology – CRYPTO 2008*, ser. Lecture Notes in Computer Science, D. Wagner, Ed., vol. 5157. Santa Barbara, CA, USA: Springer Berlin Heidelberg, Germany, Aug. 17–21, 2008, pp. 126–143.

[10] B. Hemenway, R. Ostrovsky, M. J. Strauss, and M. Wootters, "Public key locally decodable codes with short keys," in *International Workshop on Approximation Algorithms for Combinatorial Optimization*. Springer, 2011, pp. 605–615.

[11] J. Blocki, S. Kulkarni, and S. Zhou, "On locally decodable codes in resource bounded channels," in *ITC 2020: 1st Conference on Information-Theoretic Cryptography*, ser. Leibniz International Proceedings in Informatics (LIPIcs), Y. T. Kalai, A. D. Smith, and D. Wichs, Eds., vol. 163. Boston, MA, USA: Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Jun. 17–19, 2020, pp. 16:1–16:23.

[12] A. Chiesa, T. Gur, and I. Shinkar, "Relaxed locally correctable codes with nearly-linear block length and constant query complexity," in *31st Annual ACM-SIAM Symposium on Discrete Algorithms*, S. Chawla, Ed. Salt Lake City, UT, USA: ACM-SIAM, Jan. 5–8, 2020, pp. 1395–1411.

[13] T. Gur and O. Lachish, "On the power of relaxed local decoding algorithms," in *31st Annual ACM-SIAM Symposium on Discrete Algorithms*, S. Chawla, Ed. Salt Lake City, UT, USA: ACM-SIAM, Jan. 5–8, 2020, pp. 1377–1394.

[14] A. R. Block, J. Blocki, K. Cheng, E. Grigorescu, X. Li, Y. Zheng, and M. Zhu, "On Relaxed Locally Decodable Codes for Hamming and Insertion-Deletion Errors," *LIPIcs, Volume 264, CCC 2023*, vol. 264, pp. 14:1–14:25, 2023, artwork Size: 25 pages, 909793 bytes ISBN: 9783959772822 Medium: application/pdf Publisher: Schloss Dagstuhl – Leibniz-Zentrum für Informatik. [Online]. Available: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.CCC.2023.14

[15] M. H. Ameri, A. R. Block, and J. Blocki, "Memory-hard puzzles in the standard model with applications to memory-hard functions and resource-bounded locally decodable codes," in *SCN 22: 13th International Conference on Security in Communication Networks*, ser. Lecture Notes in Computer Science, C. Galdi and S. Jarecki, Eds., vol. 13409. Amalfi, Italy: Springer, Cham, Switzerland, Sep. 12–14, 2022, pp. 45–68.

[16] A. R. Block and J. Blocki, "Private and resource-bounded locally decodable codes for insertions and deletions," in *2021 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2021, pp. 1841–1846.

[17] R. L. Rivest, A. Shamir, and D. A. Wagner, "Time-lock puzzles and timed-release crypto," 1996.

[18] W. K. Pratt, J. Kane, and H. C. Andrews, "Hadamard transform image coding," *Proceedings of the IEEE*, vol. 57, no. 1, pp. 58–68, 1969.

[19] R. Ostrovsky, O. Pandey, and A. Sahai, "Private locally decodable codes," in *ICALP 2007: 34th International Colloquium on Automata, Languages and Programming*, ser. Lecture Notes in Computer Science, L. Arge, C. Cachin, T. Jurdzinski, and A. Tarlecki, Eds., vol. 4596. Wroclaw, Poland: Springer Berlin Heidelberg, Germany, Jul. 9–13, 2007, pp. 387–398.

[20] M. Christ and S. Gunn, "Pseudorandom error-correcting codes," in *Advances in Cryptology – CRYPTO 2024, Part VI*, ser. Lecture Notes in Computer Science, L. Reyzin and D. Stebila, Eds., vol. 14925. Santa Barbara, CA, USA: Springer, Cham, Switzerland, Aug. 18–22, 2024, pp. 325–347.

[21] B. Hemenway, R. Ostrovsky, M. J. Strauss, and M. Wootters, "Public Key Locally Decodable Codes with Short Keys," in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, L. A. Goldberg, K. Jansen, R. Ravi, and J. D. P. Rolim, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, vol. 6845, pp. 605–615, series Title: Lecture Notes in Computer Science. [Online]. Available: http://link.springer.com/10.1007/978-3-642-22935-0_51

[22] D. Hush and C. Scovel, "Concentration of the hypergeometric distribution," *Statistics & Probability Letters*, vol. 75, no. 2, pp. 127–132, Nov. 2005. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167715205002191

[23] E. Berlekamp, R. McEliece, and H. van Tilborg, "On the inherent intractability of certain coding problems (Corresp.)," *IEEE Transactions on Information Theory*, vol. 24, no. 3, pp. 384–386, May 1978, conference Name: IEEE Transactions on Information Theory. [Online]. Available: https://ieeexplore.ieee.org/document/1055873/?arnumber=1055873

[24] V. Weger, N. Gassner, and J. Rosenthal, "A Survey on Code-Based Cryptography," Jul. 2024, arXiv:2201.07119 [cs]. [Online]. Available: http://arxiv.org/abs/2201.07119

[25] T. Debris-Alazard and N. Resch, "Worst and average case hardness of decoding via smoothing bounds," Cryptology ePrint Archive, Report 2022/1744, 2022. [Online]. Available: https://eprint.iacr.org/2022/1744

[26] R. J. McEliece, "A public-key cryptosystem based on algebraic," *Coding Thv*, vol. 4244, pp. 114–116, 1978.

[27] E. Berlekamp, "Goppa codes," *IEEE Transactions on Information Theory*, vol. 19, no. 5, pp. 590–592, 2003.

[28] J.-C. Faugère, V. Gauthier-Umaña, A. Otmani, L. Perret, and J.-P. Tillich, "A Distinguisher for High-Rate McEliece Cryptosystems,"

IEEE Transactions on Information Theory, vol. 59, no. 10, pp. 6830–6844, Oct. 2013, conference Name: IEEE Transactions on Information Theory. [Online]. Available: https://ieeexplore.ieee.org/document/6553164/?arnumber=6553164

[29] D. J. Bernstein, T. Lange, and C. Peters, "Attacking and defending the McEliece cryptosystem," in *Post-quantum cryptography, second international workshop, PQCRYPTO 2008*, J. Buchmann and J. Ding, Eds. Cincinnati, Ohio, United States: Springer Berlin Heidelberg, Germany, Oct. 17–19, 2008, pp. 31–46.

[30] J. Leon, "A probabilistic algorithm for computing minimum weights of large error-correcting codes," *IEEE Transactions on Information Theory*, vol. 34, no. 5, pp. 1354–1359, Sep. 1988, conference Name: IEEE Transactions on Information Theory. [Online]. Available: https://ieeexplore.ieee.org/document/21270/?arnumber=21270

[31] P. J. Lee and E. F. Brickell, "An observation on the security of McEliece's public-key cryptosystem," in *Advances in Cryptology – EUROCRYPT'88*, ser. Lecture Notes in Computer Science, C. G. Günther, Ed., vol. 330. Davos, Switzerland: Springer Berlin Heidelberg, Germany, May 25–27, 1988, pp. 275–280.

[32] R. J. Lipton, "A new approach to information theory," in *Proceedings of the 11th Annual Symposium on Theoretical Aspects of Computer Science*, ser. STACS '94. Berlin, Heidelberg: Springer-Verlag, 1994, p. 699–708.

APPENDIX

A. Robust Secret Encyption from Code-based Cryptography

Code-based cryptography is a strong candidate for post-quantum cryptography due to the widely-believed intractability of code-theoretic problems related to decoding random linear codes [23], [24]. The most relevant to our work is the *average decoding problem assumption*, assuming that on average, it is hard for *any* adversary to distinguish between a codeword with a fixed number of errors and a uniformly random string.

Assumption 9 (Average Decoding Hardness [25]): There exist functions $k = k(\lambda)$, and $n = n(\lambda)$ where $k = \mathsf{poly}(\lambda)$ and $n = \theta(k)$, and $n(\lambda) \geq k(\lambda)$ such that for any probabilistic polynomial time adversary $\mathcal{A}$ playing the `ADP-Game`, there exists a negligible function ε such that for any $\lambda > 0$,

$$\left|\Pr[\texttt{ADP-Game}(\mathcal{A}, n(\lambda), k(\lambda), \lambda) = 1] - \frac{1}{2}\right| < \varepsilon(\lambda),$$

where the `ADP-Game` is defined as,

ADP-Game$(\mathcal{A}, n, k, \lambda)$

Generate $\boldsymbol{x} \xleftarrow{\$} \mathbb{F}^k, R \xleftarrow{\$} \mathbb{F}^{k\times n}$, $\boldsymbol{e} \leftarrow \mathbb{F}^n$ of hamming weight λ, and $b \xleftarrow{\$} \{0,1\}$. If $b = 0$, send $\tilde{\boldsymbol{c}} = \boldsymbol{x}R+\boldsymbol{e}$ to the adversary $\mathcal{A}$. Otherwise, if $b = 1$, send $\boldsymbol{u} \xleftarrow{\$} \mathbb{F}^n$. The adversary outputs bit $b' \in \{0,1\}$, where if $b = b'$, the output of this experiment is 1. Otherwise, the output is 0.

Robert McEliece proposed the first code-based public-key cryptosystem in 1978 [26]. His construction was based on *binary Goppa codes* and the assumption that such codes are sufficiently *indistinguishable* from random codes. We state this assumption more formally by first defining Goppa codes and the Goppa code distinguishing (GD) game referenced from previous work.

Definition 10 (Binary Goppa Code [27]): Let $g \in \mathbb{F}_{2^m}[x]$ be a polynomial of degree $\deg g \leq \lambda$ and let $L = \{\alpha_1, \dots, \alpha_n : g(\alpha_i) \neq 0\} \subset \mathbb{F}_{2^m}$ be a set of non-zero evaluation points over g. Then, a (binary) Goppa code $\mathcal{C}$ with rate $\frac{n-m\lambda}{n}$ and error tolerance λ is defined as

$$\mathcal{C} = \left\{\boldsymbol{c} = (c_1, \dots, c_n) \in \mathbb{F}^n : \sum_{i=1}^{n} \frac{c_i}{z-\alpha_i} \equiv 0 \bmod g(z)\right\}$$

Definition 11 (Goppa Code Distinguishing (GD) Game [28]):

GD-Game$(\mathcal{A}, \lambda, n, m)$

The challenger and adversary $\mathcal{A}$ interact as follows:

1) Challenger chooses at random $b \xleftarrow{\$} \{0,1\}$. If $b = 0$, then G is set to be the generator of a random linear $[n,k]$-code. Otherwise, G is set to be the generator of a random $[n,k]$-Goppa code.
2) G is given to $\mathcal{A}$, who outputs $b' \in \{0,1\}$. If $b = b'$, the output of this experiment is 1. Otherwise, the output is 0.

The claim that Goppa code generator matrices have no structure discernible from a random matrix has withstood decades of cryptanalysis and attacks which have yet to disprove this assumption for general parameters [28]–[31].

Assumption 12 (Goppa-Random Indistinguishability): There exist functions $m = m(\lambda), n = n(\lambda)$ and $k = n(\lambda) - \lambda m(\lambda)$ where $k = \mathsf{poly}(\lambda)$ and $n = \theta(k)$, and $n(\lambda) \geq k(\lambda)$ such that for for any probabilistic polynomial time adversary $\mathcal{A}$, there exists a negligible function ε such that

$$\left|\Pr[\texttt{GD-Game}(\mathcal{A}, \lambda, n, m) = 1] - \frac{1}{2}\right| < \varepsilon(\lambda, n, m).$$

The security of the McEliece cryptosystem is contingent on Assumptions 9 and 12. The cryptosystem will encode messages as codewords of a scrambled secret Goppa code with added noise, where decryption is straightforwardly undoing the scrambling and recover the message via the Goppa code decoder. More specifically, the secret key in the McEliece Cryptosystem is $\mathsf{sk} = (S, G, P)$, where G is the generator of a chosen Goppa code, P is a column permutation of G, and S is an invertible message transformation. The public key is set to be the scrambled generator $\mathsf{pk} = SGP$. The encryption scheme on input message $\boldsymbol{m}$, applies the scrambled encoding $(\boldsymbol{m}SGP)$ and adds an error vector $\boldsymbol{e}$ with weight proportional to the security parameter λ. The decryption scheme on input codeword with errors $\tilde{\boldsymbol{c}} = (\boldsymbol{m}SGP) + \boldsymbol{e}$, inverts the column permutation $\tilde{\boldsymbol{c}}P^{-1} = \boldsymbol{m}SG + \boldsymbol{e}P^{-1}$ and decodes the resulting word with the corresponding Goppa code decoder.

We construct a RSE using a similar error-obfuscation coding scheme. Our RSE construction can be interpreted as a conversion of the McEliece system to the secret/symmetric key setting with additional consideration in the amount of errors added to support the desired RSE error-tolerance. First, in the secret key setting, message scrambling matrix S and the permutation matrix P may be omitted. While these matrices were used to disassociate the public Goppa code used in encoding and the secret Goppa code used in decoding, this is unnecessary when the encoder and decoder share the same secret key. Second, the RSE robustness property requires that encodings can tolerate a δ fraction of errors. In our code-based construction, this is naturally achievable by relaxing the weight of the added error vector $\boldsymbol{e}$ by a δ fraction.

Construction 13:

$\mathsf{Gen}(1^\lambda)$

Set $\mathsf{sk} \leftarrow G$, where $G \in \mathbb{F}^{k\times n}$ be the generator of a randomly chosen $[n,k]$ Goppa Code $(\mathsf{Enc}_{\mathsf{Gop}}, \mathsf{Dec}_{\mathsf{Gop}})$ tolerant to $\lambda + \delta n$ errors.

$\mathsf{Enc}_{\mathsf{sk}}(\boldsymbol{m})$

1) Let $\boldsymbol{z} \xleftarrow{\$} \mathbb{F}^n$, where $wt(\boldsymbol{z}) = \lambda$,
2) Output $\boldsymbol{Y} \leftarrow \boldsymbol{m}G + \boldsymbol{z}$.

$\mathsf{Dec}_{\mathsf{sk}}(\widetilde{Y})$

1) From $(G,P) \leftarrow \mathsf{sk}$, compute P^{-1}.
2) Compute $\widetilde{X} = \widetilde{Y}P^{-1} = (m\widetilde{GP}+z)P^{-1} = m\widetilde{SG} + \hat{z}$, where $\hat{z} = zP^{-1}$. Note that $wt(zP^{-1}) = \lambda + \delta n$.
3) Output $\mathsf{Dec}_G(\widetilde{X})$, where Dec_G is the decoding scheme (e.g Patterson's algorithm) for the Goppa code corresponding to G.

Note that the decoding of a Goppa code, in particular Patterson's algorithm, can be done in polynomial time with the extended Euclidean algorithm.

Theorem 14: Given Assumptions 9 and 12, Construction 13 is a (n,k,δ)-RSE for any $n \geq k > 0$ and constant $\delta > 0$.

Proof: First, the robustness property is achieved by our encoding algorithm outputting a codeword Y with λ from a Goppa code that is tolerant to $\lambda + \delta n$ errors. Since the channel can induce at most δn errors, the robustness property follows dierectly from the error tolerance of the Goppa code.

Next, we show that for any $q \in \mathsf{poly}(\lambda)$, all probabilistic polynomial time adversaries $\mathcal{A}$,

$$\left|\Pr[\texttt{RSE-Game}(A,\lambda,q) = 1] - \frac{1}{2}\right| < \varepsilon(\lambda).$$

We proceed by a hybrid argument over the probability of adversary $\mathcal{A}$ winning the `RSE-Game`. We define the series of hybrids H_0 to H_2 as follows:

1) H_0: The game is played as-is.
2) H_1: Same as H_0, except in the key generation algorithm, replace the Goppa generator G to a random code generator $R \xleftarrow{\$} \mathbb{F}^{k\times n}$.
3) H_2: Same as H_1, except in encoding algorithm, replace the output in line 2 with a random word $u \xleftarrow{\$} \{0,1\}^n$.

Hybrid H_0 is indistinguishable to hybrid H_1 by Goppa-random indistinguishability in Assumption 12. More specifically, if hybrids H_0 and H_1 can be distinguished, then we can build a distinguisher for the `GD-Game` by applying the `GD-Game` challenge code (either a Goppa code or random code) to form challenge codewords for the `RSE-Game`. Note that we can claim indistinguishability across all q rounds immediately since the challenge messages are generated independently per round. Next, hybrid H_1 is indistinguishable to H_2 by average decoding hardness, which follows immediately from the encoding algorithm in hybrid H_1, outputting $\boldsymbol{x}R + \boldsymbol{e}$, where $wt(\boldsymbol{e}) = \lambda$.

In Hybrid H_2, the $b = 0$ and $b = 1$ cases are perfectly indistinguishable, and by our hybrid argument, the advantage of Hybrid H_0 is at most the advantage in Hybrid H_2 plus the negligibale advantage of the Hybrid in between. Thus, the construction satisfies RSE indistinguishability. ■

B. Detailing the One Time paLDC

Construction 15: Suppose that code $\mathcal{C} = (\mathsf{Enc}_{\mathcal{C}}, \mathsf{Dec}_{\mathcal{C}})$ is an (A,a)-code over an alphabet Σ of size q. Let $c = \log q$.

$\mathsf{Gen}(1^\lambda)$

Output $\mathsf{sk} \leftarrow (\boldsymbol{r}, \pi) \in \{0,1\}^{cAB+|\pi|}$

$\mathsf{Enc}_{\mathsf{sk}}(\boldsymbol{x})$

1) Parse $(\boldsymbol{r}, \pi) \leftarrow \mathsf{sk}$.
2) **Blocking.** Let $\boldsymbol{x} = \boldsymbol{w}_1 \circ \boldsymbol{w}_2 \circ \cdots \circ \boldsymbol{w}_B$ where each $w_s \in \Sigma^a$ for $s = 1, \ldots, B$.
3) **Block Encoding.** Encode each block s as $\boldsymbol{w}'_s = \mathsf{Enc}_{\mathcal{C}}(\boldsymbol{w}_s)$, let $\boldsymbol{x}' = \boldsymbol{w}'_1 \circ \cdots \circ \boldsymbol{w}'_B$.
4) **Permute and Mask.** Output $\pi(\boldsymbol{x}') \oplus \boldsymbol{r}$.

$\mathsf{Dec}^{\tilde{\boldsymbol{y}}}_{\mathsf{sk}}(L, R)$

Parse $(\boldsymbol{r}, \pi) \leftarrow \mathsf{sk}$. Let the interval $[L, R]$ of x bits lie in blocks $\boldsymbol{w}_s, \boldsymbol{w}_{s+1}, \ldots, \boldsymbol{w}_{s_{s+v}}$. That is, for all $i \in [L, R]$, there exists $u \in [v]$ such that x_i is a bit in $\boldsymbol{w}_{s+u}$. For each $j = s, \ldots, s+v$:

1) **Unmask.** Let $j_1, \ldots, j_\ell$, where $\ell = cA$, be the indices of the bits corresponding to $\boldsymbol{w}'_j$. Compute $\tilde{\boldsymbol{w}}'_j = (y_{\pi(j_h)} \oplus r_{\pi(j_h)})_{h \in [\ell]}$.
2) **Decode.** Apply $\mathsf{Dec}_{\mathcal{C}}(\tilde{\boldsymbol{w}}'_j)$ to obtain $\tilde{\boldsymbol{w}}_j$.

From $\tilde{\boldsymbol{w}}'_{s_1}, \ldots, \tilde{\boldsymbol{w}}'_{s_v}$ output bits corresponding to interval $[L, R]$.

Theorem 16: Suppose $\mathcal{C}$ has rate $R_{\mathcal{C}}$ and error tolerance $\delta_{\mathcal{C}}$ and $a \in \omega(\log \lambda)$. Then, Construction 15 is a $(2/R, a, \delta, 1)$-paLDC when $\delta_{\mathcal{C}} > \delta$.

Proof: For any $L, R \geq 0$ such that $R - L + 1 \geq \kappa$, $x[L, R]$ lie in blocks $w_s, w_{s+1}, \ldots, w_{s_{s+v}}$, where $v \leq \lfloor \frac{R-L+1}{ca} \rfloor + 1$. We will show that our decoding process queries at most $2(R-L+1)/R$ bits of the codeword i.e we show $\alpha \leq 2/R$.

To recover these blocks, the decoder accesses cAv bits and so,

$$\begin{aligned} \alpha &\leq \frac{cAv}{R-L+1} \\ &\leq \frac{cA\left(\lfloor \frac{R-L+1}{ca} \rfloor + 1\right)}{R-L+1} \\ &\leq \frac{1}{R} + \frac{cA}{R-L+1} \\ &\leq \frac{1}{R} + \frac{cA}{ca} \qquad (ca \leq R-L+1) \\ &= \frac{2}{R}. \end{aligned}$$

We show that the probability of an incorrect decoding $\Pr[\texttt{paLDC-Sec-Game}(\mathcal{A}, \lambda, \delta, \kappa, 1) = 1]$ is negligible by upper bounding the the probability of the event $\texttt{BAD} = \bigcup_{j \leq B} \texttt{BAD}_j$ where $\texttt{BAD}_j$ is the event that block $\boldsymbol{w}'_j$ has more than $\delta_{\mathcal{C}} A$ errors. As long as the event $\texttt{BAD}$ does not occur it is guaranteed that the local decoder will be successful in all rounds. By Lipton's theorem [32], since our encoder applies a random mask and permutation, errors are added in a uniformly random manner with probability δ. Thus, the number of errors for any given block j is hypergeometric$(AB, \delta AB, A)$ and by [21], [22] we have

$$\Pr[\texttt{BAD}_j] < 2^{\frac{-2(((\delta_{\mathcal{C}} - \delta)cA)^2 - 1)}{cA+1}},$$

which is negligible for $\delta_{\mathcal{C}} < \delta$. By a union bound over the total number of blocks, which is bounded by $\mathsf{poly}(\lambda)$, the probability, that any block fails its decoding is negligible. ■

C. *Resource-bounded* paLDC *Construction*

Lastly, we present an aLDC for resource-bounded channels with constant rate, constant amortized locality, and constant error-tolerance by applying the framework developed by Ameri et al. [15] to eliminate the requirement that the encoder and decoder have a shared secret key. The framework of Ameri et al. [15] using three building blocks: a secret key LDC, an LDC*, and cryptographic puzzles.

First, we formally introduce LDC*s, a variation on LDCs in which the entire message is recovered while making few queries to the codeword (possibly with errors).

*Definition 17 (*LDC* *[11]):* A (n, k)-code $\mathcal{C} = (\mathsf{Enc}, \mathsf{Dec})$ is an $(\ell, \delta, \varepsilon)$-LDC* if for all $\boldsymbol{x} \in \Sigma^k$, $\mathsf{Dec}^{\tilde{y}}$ with query access to word $\tilde{\boldsymbol{y}}$ δ-close to $\mathsf{Enc}(\boldsymbol{x})$,

$$\Pr[\mathsf{Dec}^{\tilde{y}} = \boldsymbol{x}] \geq (1 - \varepsilon),$$

where $\mathsf{Dec}^{\tilde{y}}$ makes at most ℓ queries to $\tilde{\boldsymbol{y}}$.
Blocki et al. crucially show that an LDC* can be constructed for locality $\ell = O(k\mathsf{poly}(1/\varepsilon))$ for arbitrarily large n with constant decoding error. Next, we formally defining cryptographic puzzles ε-hard for algorithms $\mathbb{R}$.

*Definition 18 (*Puz *[15]):* A puzzle $\mathsf{Puz} = (\mathsf{Gen}, \mathsf{Sol})$ is a $(\mathbb{R}, \varepsilon)$-hard puzzle for algorithm class $\mathbb{R}$ if there exists a polynomial t' such that for all polynomials $t > t'$ and every algorithm $\mathcal{A} \in \mathbb{R}$, there exists λ_0 such that for al $\lambda > \lambda_0$ and every $s_0, s_1 \in \{0,1\}^\lambda$, we have

$$\left| \Pr \mathcal{A}(Z_b, Z_{1-b}, s_0, s_1) = b] - \frac{1}{2} \right| \leq \varepsilon(\lambda),$$

where the probability is taken over $b \xleftarrow{\$} \{0,1\}$ and $Z_i \leftarrow \mathsf{Gen}(1^\lambda, t(\lambda), s_i)$ for $i \in \{0,1\}$.

Intuitively, a cryptographic puzzle consists of two algorithms Puz.Gen and Puz.Sol. $\mathsf{Puz.Gen}(s)$ is a randomized algorithm that takes as input a string s and outputs a puzzle Z whose solution is s i.e., $\mathsf{Puz.Gen}(Z) = s$. The security requirement is that for any adversary $A \in \mathbb{R}$ is a class $\mathbb{R}$ of resource bounded algorithms (e.g., bounded space, bounded computation depth, bounded computation) cannot solve the puzzle Z. In fact, we require that for any string s_0 and any resource bounded adversary $A \in \mathbb{R}$ the adversary A cannot even distinguish between (Z_0, s_0, s_1) and (Z_1, s_0, s_1) where s_i is a random string and $Z_i = \mathsf{Puz.Gen}(s_i)$ is a randomly generated puzzle whose corresponding solution is s_i.

We now modify the aforementioned compiler of Ameri et al. to take in a paLDC instead of an aLDC and to output a aLDC instead of a LDC for a resource-bounded channel. Additionally, we relax the definition of an aLDC to take in a consecutive range, like in paLDC. This follows naturally from the use of a paLDC to instantiate our construction. Whether resource-bounded aLDCs exist for non-consecutive queries is left as an open question. Note that the codes defined are over choice of λ values, where the message length is taken to be any polynomial $k \in \mathsf{poly}(\lambda)$.

Construction 19: Let $\mathsf{paLDC}.(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be a paLDC that is a $(n_\mathsf{P}, k_\mathsf{P})$-code, let $\mathsf{LDC}^*.(\mathsf{Enc}, \mathsf{Dec})$ be a LDC*that is a (n_*, k_*)-code, and let $\mathsf{Puz}.(\mathsf{Gen}, \mathsf{Sol})$ be a $(\mathbb{R}, \varepsilon)$-hard puzzle. Let t' be the polynomial guaranteed by Definition 18. Then, for any fixed $\lambda \in \mathbb{N}$, we construct an aLDC $(\mathsf{Enc}, \mathsf{Dec})$ as the following algorithms.

$\mathsf{Enc}(\boldsymbol{x})$

1) Sample random seed $\boldsymbol{s} \xleftarrow{\$} \{0,1\}^{k_\mathsf{P}}$.
2) Choose polynomial $t > t'$ and compute $\boldsymbol{Z} \leftarrow \mathsf{Puz.Gen}(1^\lambda, t(\lambda), s)$ where $\boldsymbol{Z} \in \{0,1\}^{k_*}$.
3) Set $Y_* \leftarrow \mathsf{LDC}^*.\mathsf{Enc}(\boldsymbol{Z})$.
4) Set $\mathsf{sk} \leftarrow \mathsf{paLDC.Gen}(1^\lambda; \boldsymbol{s})$ i.e. we explicitly instantiate paLDC.Gen with random seed $\boldsymbol{s}$.
5) Set $Y_\mathsf{P} \leftarrow \mathsf{paLDC.Enc}_\mathsf{sk}(\boldsymbol{x})$
6) Output $Y_* \circ Y_\mathsf{P}$.

$\mathsf{Dec}^{\widetilde{Y_\mathsf{P}} \circ \widetilde{Y_*}}(L, R)$

1) Decode $\boldsymbol{Z} \leftarrow \mathsf{LDC}^*.\mathsf{Dec}^{\widetilde{Y_*}}$.
2) Compute $\boldsymbol{s} \leftarrow \mathsf{Puz.Sol}(\boldsymbol{Z})$
3) Compute $\mathsf{sk} \leftarrow \mathsf{paLDC.Gen}(1^\lambda; \boldsymbol{s})$
4) Output $\mathsf{paLDC.Dec}_\mathsf{sk}^{\widetilde{Y_\mathsf{P}}}(L, R)$

Theorem 20: Suppose Construction 19 is instantiated with an $(\alpha_\mathsf{P}, \kappa_\mathsf{P}, \delta_\mathsf{P}, \varepsilon_\mathsf{P}, q)$-paLDC, $(\ell_*, \delta_*, \varepsilon_*)$-LDC*, and a $(\mathbb{R}, \varepsilon_\mathsf{Puz})$-hard puzzle Puz. Then, Construction 19 is a (n, k)-code that is a $(\alpha, \kappa, \delta, \varepsilon)$-aLDC with

$$\begin{aligned}
n &= n_\mathsf{P} + n_*, \\
k &= k_\mathsf{P}, \\
\alpha &= \alpha_p + \frac{\ell_*}{\kappa_\mathsf{P}}, \\
\kappa &= \kappa_\mathsf{P}, \\
\delta &= (1/n) \times \min\{\delta_* n_*, \delta_\mathsf{P} n_\mathsf{P}\}, \\
\varepsilon &= ((1 - \varepsilon_*)\varepsilon_\mathsf{P} + 2\varepsilon_\mathsf{Puz})/\varepsilon_*,
\end{aligned}$$

when the adversarial channel $\mathcal{A} \in \mathbb{R}$.

Proof: The decoder given $L, R \geq 0$ such that $R - L + 1 \geq \kappa_\mathsf{P}$, queries the word $\tilde{Y_*} \circ \tilde{Y_\mathsf{P}}$ in two steps. First, it performs queries for $\mathsf{LDC}^*.\mathsf{Dec}^{\tilde{Y_*}}$ to recover the puzzle $\boldsymbol{Z}$, and after solving the puzzle to generate secret key sk, it performs queries for $\mathsf{paLDC.Dec}_\mathsf{sk}^{\tilde{Y_\mathsf{P}}}(L, R)$. The total number of queries is at most $\ell_* + (R - L + 1)\alpha_\mathsf{P}$ so α can be derived as

$$\alpha(R-L+1) \leq \ell_* + (R-L+1)\alpha_\mathsf{P} \implies \alpha \leq \alpha_\mathsf{P} + \ell_*/\kappa_\mathsf{P}.$$

Next, the error tolerance δ can be derived as the weighted minimum error-tolerance between the LDC* and paLDC codewords, $\delta = (1/n) \times$

$\min\{\delta_* n_*, \delta_{\mathsf{P}} n_{\mathsf{P}}\}$. A channel can choose either Y_{P} or Y_* to place all δn errors, so the fraction of errors tolerated is the minimum of the error tolerances of the chosen LDC^* and paLDC.

The remainder of the proof of security and decoding error follows the proof of Theorem 6.8 of [15] with minor changes for notational differences. We repeat it for the sake of completeness. Suppose there exists an algorithm/adversarial channel $\mathcal{A} \in \mathbb{R}$ that, given the hard puzzle Puz, can cause a decoding error for some chosen range $[L, R]$ with some non-negligible probability $\varepsilon(\lambda)$ for some $\lambda \in \mathbb{N}$. Then, we can construct an adversary $\mathcal{B} \in \mathbb{R}$ breaking security of the $(\mathbb{R}, \varepsilon_{\mathsf{Puz}})$-hard puzzle by a two-phase hybrid argument. In the first phase, we define two encoders Enc_0 and Enc_1, where Enc_0 is equal to the original encoder in Construction 19 and Enc_1 is defined to take in the secret key sk (rather than generating it as in Enc_0) as follows.

$\mathsf{Enc}_1(\boldsymbol{x}, \mathsf{sk})$

1) Sample random seed $\boldsymbol{s} \xleftarrow{\$} \{0,1\}^{k_{\mathsf{P}}}$.
2) Choose polynomial $t > t'$ and compute $\boldsymbol{Z} \leftarrow \mathsf{Puz.Gen}(1^\lambda, t(\lambda), s)$ where $\boldsymbol{Z} \in \{0,1\}^{k_*}$.
3) Set $Y_* \leftarrow \mathsf{LDC}^*.\mathsf{Enc}(\boldsymbol{Z})$.
4) Set $Y_{\mathsf{P}} \leftarrow \mathsf{paLDC.Enc}_{\mathsf{sk}}(\boldsymbol{x})$
5) Output $Y_* \circ Y_{\mathsf{P}}$.

We proceed with constructing the two-phase hybrid distinguisher $(\mathcal{D}_1, \mathcal{D}_2)$. First, $\mathcal{D}_1$ is given as input message $\boldsymbol{x}$ and access to encoders Enc_0 and Enc_1.

$\mathcal{D}_1(\boldsymbol{x})$

1) Compute $b \xleftarrow{\$} \{0,1\}$ and $\mathsf{sk}_1 \leftarrow \mathsf{paLDC.Gen}(1^\lambda)$.
2) If $b = 0$, output $\boldsymbol{Y}_b \leftarrow \mathsf{Enc}_0(\boldsymbol{x})$. Otherwise, if $b = 1$, output $\boldsymbol{Y}_b \leftarrow \mathsf{Enc}_1(\boldsymbol{x}, \mathsf{sk})$.

Denote the secret key used in Enc_b as sk_b, where $b \in \{0,1\}$. sk_b is either generated by the distinguisher (when $b = 1$) or the encoding algorithm Enc_0 (when $b = 0$). The output of $\mathcal{D}_1(\boldsymbol{x})$ is given to the adversarial channel $\mathcal{A}$, who outputs $\boldsymbol{Y}'_b = \boldsymbol{Y}'_{\mathsf{P},b} \circ Y'_*$. Then, in phase two, $\mathcal{D}_2$ is given as input the message $\boldsymbol{x}$, the secret key sk_b, and the corrupt paLDC codeword $\boldsymbol{Y}'_{\mathsf{P},b}$, where b is the bit computed by $\mathcal{D}_1(\boldsymbol{x})$.

$\mathcal{D}_2(\boldsymbol{x}, \mathsf{sk}_b, \boldsymbol{Y}'_{\mathsf{P},b})$

1) Sample $L, R \xleftarrow{\$} [k]$ such that $L \le R$ and $R - L + 1 \ge \kappa$.
2) Compute $\boldsymbol{x}'_i \leftarrow \mathsf{paLDC.Dec}_{\mathsf{sk}_b}^{\boldsymbol{Y}'_{\mathsf{P},b}}(L, R)$.
3) Output $b' = 0$ if $\boldsymbol{x}_i \neq \boldsymbol{x}'_i$, otherwise output $b' = 1$.

Now, we give the two-phase distinguisher which breaks the $(\mathbb{R}, \varepsilon_{\mathsf{Puz}})$-hard puzzle. For puzzle solutions s_0, s_1 we construct an adversary $\mathcal{B} \in \mathbb{R}$ which distinguishes $(\boldsymbol{Z}_b, \boldsymbol{Z}_{1-b}, s_0, s_1)$ with probability at least ε' for $b \xleftarrow{\$} \{0,1\}$. Fix a message $\boldsymbol{x}$ and $\lambda \in \mathbb{N}$.

$\mathcal{B}(\boldsymbol{Z}_b, \boldsymbol{Z}_{1-b}, s_0, s_1)$

1) **Encoding.**
   a) Generate $\mathsf{sk} \leftarrow \mathsf{paLDC.Gen}(1^\lambda; s_0)$.
   b) Set $Y_* \leftarrow \mathsf{LDC}^*.\mathsf{Enc}(\boldsymbol{Z}_b)$.
   c) Set $Y_{\mathsf{P}} \leftarrow \mathsf{paLDC.Enc}_{\mathsf{sk}}(\boldsymbol{x})$.
   d) Set $\boldsymbol{Y} = Y_* \circ Y_{\mathsf{P}}$.
2) Give $(\boldsymbol{x}, \boldsymbol{Y}, \delta, \varepsilon, k, n)$ to $\mathcal{A}$ to obtain $\boldsymbol{Y}'$. Set Y'_{P} as the substring corresponding to the corruption of Y_{P} above.
3) Compute $x'_L, \ldots, x'_R \leftarrow \mathsf{paLDC.Dec}_{\mathsf{sk}}^{Y'_{\mathsf{P}}}$ and if there exists $x'_i \neq x_i$ for some $i \in [L, R]$, output $b' = 0$. Else output $b' = 1$.

First, by assumption, the adversary $\mathcal{B} \in \mathbb{R}$ since each of its subroutines $\mathcal{A}, \mathsf{paLDC.Enc}, \mathsf{paLDC.Dec}, \mathsf{LDC}^* \in \mathbb{R}$. We claim that $\mathcal{B}$ distinguishes $(\boldsymbol{Z}_b, \boldsymbol{Z}_{1-b}, s_0, s_1)$ with noticeable advantage. First, observe that sk is always generated by $\mathsf{paLDC.Gen}(1^\lambda, s_0)$. Next, for $b = 1$, Y_* encodes puzzle $\boldsymbol{Z}_1$ and the secret key sk is unrelated to the solution s_1 of puzzle $\boldsymbol{Z}_1$. Since in this case, sk and Y_* are uncorrelated, $\mathcal{A}$ causes a decoding error with probability ε_{P} where $\boldsymbol{Y}'$ is at most δn distance away from $\boldsymbol{Y}$.

In the case that $b = 0$, puzzle $\boldsymbol{Z}_0$ is encoded as Y_* with solution s_0 that is used to generate sk. Thus, in this case, the probability that the decoder outputs at least one incorrect x_i is at least the advantage of $\mathcal{A}$, ε.

$\mathcal{B}$ outputs $b' = b$ when $b = 0$ with probability $\varepsilon\varepsilon_*(1/k)$ by the argument above. For $b = 1$, this probability is at least $1 - \varepsilon_{\mathsf{P}}\varepsilon_*$ since the probability that $b' = 0$ in this case is at most $\varepsilon_{\mathsf{P}}\varepsilon_*$. Thus, we have that

$$\Pr[\mathcal{B}(\boldsymbol{Z}_b, \boldsymbol{Z}_{1-b}, s_0, s_1) = b] - \frac{1}{2} \geq \frac{1}{2}\left(\varepsilon\varepsilon_*(1/k) - \varepsilon_{\mathsf{P}}(1 - \varepsilon_*)\right),$$

which is non-negligible, contradicting the security of the puzzle Puz. ■

Explicitly, we can achieve constant decoding error ε, constant rate R, constant amortized locality α, and constant error-tolerance δ by using the (one-time) paLDC construction from Theorem 16 and an LDC^* with $n_* \sim n_{\mathsf{P}}$. Note that $n_* = \theta(n_{\mathsf{P}})$ is necessary for simultaneous optimally constant parameters. We need $n_* = \Omega(n_{\mathsf{P}})$ for constant error tolerance, and, at the same time, the rate $R = k_{\mathsf{P}}/(n_{\mathsf{P}} + n_*)$ is constant for $n_* = O(n_{\mathsf{P}})$.