Handling Correlated Errors: Hardness of LWE in the Exponent

Luke Demarest* Benjamin Fuller† Alexander Russell‡

October 17, 2018

Abstract

The hardness of decoding random linear codes with errors is a complexity-theoretic assumption with broad applications to cryptography. In contrast, Reed-Solomon codes permit efficient decoding in many representations. Despite this, a result of Peikert (TCC 2006) proves that in groups where discrete log is hard it is difficult to perform Reed-Solomon error correction if each symbol is written in the exponent. We bring these two lines of work together, examining hardness of decoding random linear codes in the exponent.

Our main result is a pair of theorems that show hardness of decoding random linear codes in both the generic group model and the standard model. In the generic group model our analysis can be carried out for a quite general family of distributions for (the support of) the error terms. We show hardness of decoding as long as every subset of group elements (of size at least the dimension of the code) has an overwhelming probability of at least one random error. This family includes error distributions whose symbols are correlated. Our results in the standard model show hardness of decoding random linear codes with a uniform input point. These results improve on a result of Peikert (TCC 2006) who considered the problem for Reed-Solomon codes.

We explore two applications of these results. First, we construct a reusable fuzzy extractor with storage independent of the number of errors to be corrected. This construction unifies the strengths of two prior constructions (Fuller, Meng, and Reyzin, Asiacrypt 2013) and (Canetti et al., Eurocrypt 2016). Second, we show how to build virtual black-box obfuscation for a class of functionality known as *pattern matching*. Recently, Bishop et al. (Crypto 2018) constructed a scheme based on codes in the exponent and showed security for uniform inputs. We show the same construction is secure for more distributions. The security arguments of both applications rely on distributions drawn from some physical process which are best modeled by distributions with correlated bits.

Keywords Learning with errors; error-correction; generic group model; fuzzy extractors; pattern-matching obfuscation

1 Introduction

We study the problem of decoding random linear codes "in the exponent," and explore direct applications to constructing fuzzy extractors [DORS08] and pattern matching obfuscation [BKM+18] (we review these applications in detail in Section 1.2 after introducing our technical results). One notable conclusion of our study is that such decoding remains hard for quite general error distributions; in particular, we obtain

*Email: luke.johnson@uconn.edu. University of Connecticut.

†Email: benjamin.fuller@uconn.edu. University of Connecticut.

‡Email: acr@uconn.edu University of Connecticut.

results for error distributions for which standard *learning with errors* (LWE) is not known to be difficult. This generality is critical for our applications.

Our approach has two direct motivations: (i.) the flexibility and broad applicability of LWE to cryptography and (ii.) the striking fact that decoding Reed-Solomon codes is difficult in the exponent even though efficient algorithms exist when symbols are in the clear [Pei06]. This suggests that the problem of decoding random linear codes in the exponent might provide a cryptographic tool with superior flexibility. One might expect moving a random code to the exponent to similarly impart augmented intractability. We discuss these two motivations in more detail below.

Hardness of Decoding Random Linear Codes. In the last twenty years, the hardness of decoding error correcting codes has become a core cryptographic assumption. Indeed, hardness of decoding random linear codes can serve as the underlying assumption for almost every cryptographic primitive [BMvT78, Reg05, BV14, GVW15, WZ17]. The relevant cryptographic assumption is called LWE [Reg10]. The decision version asks an adversary to distinguish an encoded codeword with errors $\mathbf{Ax}+\mathbf{e}$ from a uniformly selected vector, where $\mathbf{x}$ describes a message, $\mathbf{A}$ is a random (public) matrix, and $\mathbf{e}$ is an error term. Here all operations to be performed in the finite field $\mathbb{F}_q$. While this problem is easy in the continuous setting (using linear regression), it appears hard in finite fields. In fact, for particular distributions of $\mathbf{e}$ it is possible to show that any adversary that solves LWE implies an adversary that can approximately solve worst case lattice problems (`GapSVP` [BLP+13] or `SIVP` [Reg05]). While there is some flexibility in the distribution of $\mathbf{e}$ (e.g., a discretized Gaussian [Reg05], a uniform interval [DMQ13], or a binomial distribution [MP13]), known reductions require each symbol of $\mathbf{e}$ to be independently distributed.

Structured Codes in the Exponent. Structured codes also have far-reaching applications in cryptography. Throughout this work we consider codes of length n that encode k message symbols and are designed to correct t errors. An example is the Reed-Solomon code which treats a message $m_1, ..., m_k$ as coefficients of a degree k polynomial $p(\cdot)$. The associated codeword is the polynomial evaluated at points $1, ..., n$: $p(1), \ldots, p(n)$. Using Lagrangian interpolation it is possible to reconstruct the polynomial from any k points.

An example of how structured codes can augment schemes is threshold public key encryption [Des92]. Recall El Gamal encryption [ElG85]: with the secret key x, encryptions have the form $(g^y, m \cdot g^{xy})$. We suppose now that n users have evaluations of a polynomial $p(\cdot)$ with intercept x and other coefficients randomly chosen. Lagrangian interpolation asserts that for any set $\mathcal{I}$ of size at least k there exists $\lambda_1, ..., \lambda_k$ such that $x = \sum_{i=1}^{k} \lambda_1 x_{i_1}$. Since this operation is entirely linear users can decrypt without revealing their shares by announcing $g^{\lambda_i x_i y}$ and computing:

$$\frac{m \cdot g^{xy}}{\prod_{i=1}^{n} g^{\lambda_i x_i y}} = \frac{m \cdot g^{xy}}{g^{\sum_{i=1}^{n} \lambda_i x_i y}} = m.$$

This scheme is private against adversaries that know less than k shares and is correct against adversaries that keep at most $n - k$ parties from participating in reconstruction. However, it is also possible to correct errors for Reed-Solomon codes: indeed, the Berlekamp-Welch [WB86] algorithm can efficiently correct up to $t = (n - k + 1)/2$ errors. Suppose there is an adversary that intentionally submits incorrect shares $z_i \neq g^{\lambda_i x_i y}$ during reconstruction. A natural question is whether Reed-Solomon error-correction can be applied to ensure correctness against this adversary. As it happens, the Berlekamp-Welch decoding algorithm critically relies on nonlinear operations, so its not clear how to execute this algorithm "in the exponent." Peikert showed this is, in some sense, unavoidable with two results [Pei06]:

1. If balls of radius t around codewords (the set of all degree k polynomials) cover a noticeable fraction of the metric space, an algorithm that corrects t errors implies a solver for the discrete logarithm problem. Importantly, this algorithm only has to find some codeword within distance t (and not, necessarily, the closest codeword). This is known as bounded distance decoding [CW07] in contrast to the more traditional unique decoding. In order for balls around codewords to cover a noticeable fraction of the space, one needs to consider an algorithm that corrects almost $t = n - k$ errors. Note that for a polynomial of degree k performing bounded distance decoding in the exponent for distance $n - k$ is easy: One picks k points and interpolates the rest of the points which can be done linearly and in the exponent. We stress this result does not rule out algorithms which correct fewer errors.
2. In the generic group model [Sho97], adversaries are limited to linear combinations of group elements. In particular, the adversary's algorithm must work for an arbitrary representation of the group. In this model, Peikert showed that given $g^{\mathbf{c}+\mathbf{e}}$, where $\mathbf{c}$ is a codeword and $\mathbf{e}$ is a random vector with t nonzero values, it is hard to find $g^{\mathbf{c}}$. In this model, no assumption is necessary about the density of codewords in the metric space. The main requirements for this theorem are (i.) that t errors are uniformly distributed and (ii.) that the product of the dimension and number of errors is large enough (namely, that $tk = \omega(n \log n)$).

Peikert's work presented the above as negative results. However, these results can be seen as introducing a new type of hardness assumption. The goal of this work is to combine the above two lines of work and ask:

> Does placing a random linear code in the exponent of a group amplify the hardness of decoding?

1.1 Our Contribution: Hardness of Decoding Random Linear Codes in the Exponent

We consider error distributions $\mathbf{e}$ that are the product of two components: $\mathbf{e}'_i$, a vector of independently random group elements, and $\mathbf{s}$, a binary selection vector that determines the location of the errors. That is, $e_i = e'_i \cdot s_i$. Our goal is to support as many distributions S over the binary vector $\mathbf{s}$ as possible. In Subsection 1.2, we describe two applications which make particular demands on the error distribution. Importantly, we make no assumption about the bits of the selection vector being independent, and thus the positions i where $e_i = 0$ are potentially correlated.

A Necessary and Sufficient Condition. Peikert's result in the generic group model only holds if the product of errors and the size of the message is big enough (when $tk = \omega(n \log n)$). This is because an adversary can repeatedly try to find subsets of size k without any errors and perform a linear operation to recover the original codeword. This algorithm was formalized by Canetti and Goldwasser [CG99] and succeeds when $tk = \Theta(n \log n)$. In the setting where nonzero errors are uniform, a necessary condition for security is that each subset of size k has an overwhelming probability of including a nonzero error. That is, for each subset $\mathcal{I}$ of size at least k, the min-entropy of S restricted to $\mathcal{I}$ is at least some α that is superlogarithmic in the security parameter: $\mathrm{H}_\infty(S_\mathcal{I}) \geq \alpha$. We say that a distribution with this property has (α, k)-*entropy subsets* (see Definition 1).

Our main result is in the generic group model. We show that the (α, k)-entropy subset condition is a *sufficient* condition for hardness of decoding a random linear code (Theorem 1).

Standard Model Results. We also provide hardness results in the standard model, however these results require S to have independent symbols, with S possessing t randomly chosen nonzero positions.

We show this result for both random linear codes (Theorem 6) and Reed-Solomon codes (Theorem 7). Both results improve parameters of Peikert's result [Pei06, Theorem 3.1]. As stated, these arguments require that a random point lies close to a codeword with noticeable probability. As q increases this probability decreases but discrete log becomes harder, creating a tension between these parameters. Peikert's result requires that $q \leq \binom{n}{k+1}/n^2$. In our application to the fuzzy extractors we consider small k for which $k = \omega(\log \lambda)$. This means that the upper bound on q may be just superpolynomial. Our results allow q to grow more quickly, improving the bound by a modest factor of n^2 (requiring that $q \leq \binom{n}{k+1}$).

Theorems 6 and 7 consider an adversary that performs error correction: given $g^{\mathbf{y}}$ it returns $g^{\mathbf{z}}$ where the distance between $\mathsf{dis}(\mathbf{y}, \mathbf{z}) \leq t$ and $g^{\mathbf{z}}$ is a codeword. Recently, Fuchsbauer et al. [FKL18] introduced the algebraic group model which is weaker than the generic group model. From an input $g^{\mathbf{y}}$, an *algebraic* adversary produces a solution $g^{\mathbf{z}}$ along with a matrix $\vec{\Lambda}$ such that $g^{\mathbf{z}} = g^{\vec{\Lambda}\mathbf{y}}$. The model is weaker than the generic group model as the adversary is allowed to see the elements $g^{\mathbf{y}}$ before creating $\vec{\Lambda}$. Any standard model adversary that decodes a linear code implies an algebraic adversary. One can find k indices where $g^{\mathbf{z}_i} = g^{\mathbf{y}_i}$. One then uses the *linear* decoding (from these indices) and encoding procedures of the code to find the coefficients such that $g^{\mathbf{z}} = g^{\Lambda\mathbf{y}}$.

1.2 Applications

Fuzzy Extractors. A fuzzy extractor derives stables keys from a noisy source of entropy [DORS08]. Formally, a fuzzy extractor is a pair of algorithms:

- $\mathsf{Gen}(\mathbf{w})$ (generate), which takes an initial reading of a noisy source $\mathbf{w}$ and outputs a cryptographic key key and a public value pub.
- $\mathsf{Rep}(\mathbf{w}', \mathsf{pub})$ (reproduce), which takes a subsequent reading $\mathbf{w}'$ and outputs key if $\mathbf{w}$ and $\mathbf{w}'$ are within distance t. The security guarantee is that key should be pseudorandom conditioned on pub.

Fuzzy extractors secure distributions from nature such as biometric and physical uncloneable functions, so it is prudent to minimize assumptions about these distributions. For example, the Iriscode transform is estimated to have 249 bits of entropy out of a 2048 bit string [Dau04] (see discussion in [FRS16]). Importantly bits of $\mathbf{w}$ are *correlated*.

Fuller et al. [FMR13] proposed a fuzzy extractor where the reading $\mathbf{w}$ served as the error vector for an LWE instance, that is $\mathsf{pub} = \mathbf{A}\mathbf{x} + \mathbf{w}$.[1] The Rep algorithm performs "guess and check" on subsets $\mathcal{I} \subseteq \{1, ..., |\mathbf{w}|\}$ decoded as

$$\mathbf{x} = \mathbf{A}_{\mathcal{I}}^{-1}\mathsf{pub}_{\mathcal{I}} - \mathbf{w}'_{\mathcal{I}} = \mathbf{x}\mathbf{A}_{\mathcal{I}}^{-1}(\mathbf{w} - \mathbf{w}')_{\mathcal{I}}.$$

This decoding succeeds if $\mathbf{w}_{\mathcal{I}} = \mathbf{w}'_{\mathcal{I}}$. To achieve error tolerance, multiple independent sets $\mathcal{I}$ are sampled and checked. Importantly, this decoding algorithm selects a subset and is then entirely linear. To ensure hardness of the underlying lattice problem, the construction required: (i.) the dimension of $\mathbf{x}$ to be a constant fraction the length of w and (ii.) for $\mathbf{w}$ to be uniformly distributed.[2] This limited error tolerance to at most $t = O(\log |w|)$.

Canetti et al. [CFP+16] presented a fuzzy extractor that explicitly placed specific subsets in a digital locker [CD08]. Digital lockers can be constructed using exponentiation in a Diffie-Hellman group [BC10].

[1] The cryptographic key key is part of the $\mathbf{x}$ vector. Akavia, Goldwasser, and Vaikuntanathan showed when LWE is sufficiently hard parts of $\mathbf{x}$ are hardcore [AGV09].

[2] This distribution was shown to be hard for LWE decoding by Döttling and Müller–Quade [DMQ13]. More generally, the construction is secure for any error distribution where LWE is hard.

This construction sampled multiple subsets $w_{\mathcal{I}_j}$ and locked the same key in a digital locker secured with $w_{\mathcal{I}_j}$. Roughly,

$$\mathsf{pub} = \{r, r^{w_{\mathcal{I}_j}} \cdot \mathsf{key}, \mathcal{I}_j\}_j.$$

Decoding consisted of trying to open each digital locker. Digital lockers improved allowable error tolerance to $t = o(|w|)$. However, this construction explicitly writes each subset to be tested. To achieve meaningful error tolerance for an actual biometric millions of these lockers are required [SSF18]. Canetti et al.'s construction is also *reusable*, allow a correlated versions of w to be used to derive multiple keys.

We introduce a new fuzzy extractor that places a random linear code in the exponent:

$$\mathsf{pub} = \left\{ \mathsf{pub}_i = \begin{cases} g^{\mathbf{A}_i \mathbf{x}} & w_i = 0 \\ g^{r_i}, r_i \leftarrow \mathbb{Z}_p^* & w_i = 1 \end{cases} \right\}_{i=0}^{|w|-1}.$$

Decoding proceeds as in the previous constructions, randomly selecting subsets $\mathcal{I}_j$ and hoping they have no errors. This construction is secure for all distributions where it is hard to decode a random linear code in the exponent with random errors placed according to the 1s of w. (Errors are actually placed according to when the bits of $\mathbf{w}$ match a uniform random string.) If the construction is instantiated with a random linear code of small dimension $|\mathbf{x}| = \omega(\log \lambda)$ error tolerance of $t = o(|w|)$ is possible (as in Canetti et al. [CFP+16]). If independent generators are used each time that Gen is run, this construction is also reusable, allowing multiple enrollments of a single physical source.

Pattern Matching Obfuscation. In a recent work, Bishop et al. [BKM+18] show how to obfuscate a pattern $\mathbf{v}$ where each $v_i \in \{0, 1, \perp\}$ indicating that the bit v_i should match 0, 1 or either value. The goal is to allow a user to check for input string y, if y and v are the same on all non-wildcard positions. Their construction was stated for Reed-Solomon codes but works for any linear code. We state the construction for a random linear code: Let $|\mathbf{v}| = n$ and assume $\mathbf{A} \in (\mathbb{Z}_p)^{2n \times n}$. Then for a random $\mathbf{x}$ the construction outputs the following obfuscation:[3]

$$\mathcal{O}_w = \left\{ o_i = \begin{cases} (g^{\mathbf{A}_{2i}\mathbf{x}}, g^{r_{2i+1}}), r_{2i+1} \leftarrow \mathbb{Z}_p^* & v_i = 1 \\ (g^{r_{2i}}, g^{\mathbf{A}_{2i+1}\mathbf{x}}), r_{2i} \leftarrow \mathbb{Z}_p^* & v_i = 0 \\ (g^{\mathbf{A}_{2i}\mathbf{x}}, g^{\mathbf{A}_{2i+1}\mathbf{x}}) & v_i = \perp \end{cases} \right\}_{i=0}^{|v|-1}.$$

Bishop et al. prove security of the scheme in the generic group model. Intuitively, their argument rests on two facts: (i.) its hard to isolate wildcard positions where both values can be used to find $\mathbf{x}$ and (ii.) for nonwildcard positions its hard to pick a set without including errors. Their analysis focuses on allowing a large number of randomly placed wildcards with the uniform distribution for nonwildcard bits of $\mathbf{v}$. Most applications of string matching are on nonuniform and correlated values such as human language. We show the same construction is secure for more distributions over $\mathbf{v}$. First, we define an auxiliary variable $\mathbf{s}$ of length $2n$ that describes the placement of errors as follows:

$$s_i = \begin{cases} 10 & \text{if } v_i = 1, \\ 01 & \text{if } v_i = 0, \\ 00 & \text{if } v_i = \perp. \end{cases}$$

[3]Bishop et al. state their construction where $\mathbf{x}_0 = 0$ to allow the user to check whether they matched the pattern. In this description, we allow the user to get out a key contained in $g^{\mathbf{x}_0}$ when they are correct.

We show it is sufficient for the string S to have (α, n)-entropy subsets for a superlogarithmic α. (Note this condition is also necessary for security using the algorithm of Canetti and Goldwasser.) This class of distributions only requires each subset of bits to be unpredictable. In human language, subsets of bits do have this property [Sha51, BPM+92, MZ11].

Concurrent Work. In concurrent work Bartusek, Lepoint, Ma, and Zhandry [BLMZ18] present two contributions of interest to this work. They consider the pattern matching obfuscation application. Their first contribution raises the upper bound on the number of wildcards in [BKM+18] from $w < 0.774n$ to $w < n - \omega(\log n)$ using a new dual form of analysis. Their analysis still considers the uniform distribution over nonwildcard positions. Thus, our analysis expands the provably secure distributions over $\mathbf{v}$. Their second contribution considers random linear codes not in the exponent, they use a modified version of the Random Linear Code (RLC) assumption defined in [IPS09]. They prove for some structured error distributions hardness of both search and decision problems. Constructions that are secure outside the group (their construction being one) are immediately hard in the generic group model as well. Importantly, their analysis relies on the adversary receiving only $2n$ dimensions of the code and would not apply for our fuzzy extractor application.

Organization. The remainder of the paper is organized as follows, Section 2 covers definitions and preliminaries, Section 3 presents our main theorem on hardness of decoding random linear codes in the generic group model. Sections 4 and 5 describe our applications to fuzzy extractors and pattern matching obfuscation respectively. Finally Section 6 shows hardness of decoding high entropy errors in the standard model.

2 Definitions

For random variables X_i over some alphabet $\mathcal{Z}$ we denote the tuple by $X = (X_1, ..., X_n)$. For a set of indices J, X_J denotes the restriction of X to the indices in J. For a vector $\mathbf{v}$ we denote the ith entry v_i. The *min-entropy* of X is $\mathrm{H}_\infty(X) = -\log(\max_x \Pr[X = x])$. The *average (conditional)* min-entropy of X given Y is $\tilde{\mathrm{H}}_\infty(X \mid Y) = -\log(\mathbb{E}_{y \in Y} \max_x \Pr[X = x \mid Y = y])$ [DORS08, Section 2.4].

The *statistical distance* between random variables X and Y with the same domain is

$$\Delta(X, Y) = \frac{1}{2} \sum_x |\Pr[X = x] - \Pr[Y = x]|.$$

For a distinguisher D we define the *computational distance* between X and Y as

$$\delta^D(X, Y) = |\mathbb{E}[D(X)] - \mathbb{E}[D(Y)]|$$

and extend this notion to a class of distinguishers $\mathcal{D}$ by taking the maximum over all distinguishers $D \in \mathcal{D}$. We denote by $\mathcal{D}_s$ the class of randomized circuits which output a single bit and have size at most s.

For a metric space $(\mathcal{M}, \mathsf{dis})$, the *(closed) ball of radius t around x* is the set of all points within radius t, that is, $B_t(x) = \{y \mid \mathsf{dis}(x, y) \leq t\}$. If the size of a ball in a metric space does not depend on x, we denote by $\mathsf{Vol}(t)$ the size of a ball of radius t. We consider the Hamming metric. Let $\mathcal{Z}$ be a finite set and consider vectors in $\mathcal{Z}^n$, then $\mathsf{dis}(x, y) = |\{i \mid x_i \neq y_i\}|$. For this metric, we denote volume as $\mathsf{Vol}(n, t, |\mathcal{Z}|)$ and $\mathsf{Vol}(n, t, \mathcal{Z}) = \sum_{i=0}^{t} \binom{n}{i} (|\mathcal{Z}| - 1)^i$. U_n denotes the uniformly distributed random variable on $\{0, 1\}^n$. Unless otherwise noted logarithms are base 2. Usually, we use capitalized letters for random variables and corresponding lowercase letters for their samples.

Error Selection Distribution Our core technical result considers a distribution $\mathbf{e}$ where each component is the product of a random value and a binary value s_i where $\mathbf{s}$ is drawn from some distribution in "nature." Our security guarantees will require that any subset of positions $\mathcal{I}$ is likely to have a nonzero value of $\mathbf{s}_\mathcal{I}$ with high probability. Formally, we call such a distribution S a source with (α, k)-*entropy subsets*:

Definition 1. *Let a source $S = S_1, \ldots, S_n$ consist of n-bit binary strings. For some parameters k, α we say that the source S is has (α, k)-**entropy subsets** if $\mathrm{H}_\infty(S_{j_1}, \ldots, S_{j_k}) \geq \alpha$ for any $1 \leq j_1, \ldots, j_k \leq n$.*

3 Hardness of Decoding in the Generic Group Model

Our first technical result concerns the hardness of decoding a random linear code when symbols are given in the exponent. We adopt the *generic group model* [Sho97] which reflects computation with (typically cyclic) groups in an idealized setting where the representation of group elements permits efficient computation of the group law, but leaks no other useful information. Equivalently, the model captures the power of adversaries whose strategy for a cryptographic game does not depend on the representation of group elements. The model calls for a function which assigns to each group element g_i a random bitstring (of a sufficient length to prevent collisions). We assume this mapping is one-to-one throughout and denote it by σ. In a typical security game, the adversary is initially given access to some starting "handles" $\sigma(g_i)$ (for example, a generator for the group along with some further handles representing group elements with particular cryptographic relevance). The adversary is allowed to ask an oracle $\mathcal{O}$ "group law" queries of the form $\sigma(g_i) \pm \sigma(g_j)$ for any two previously observed handles $\sigma(g_i)$ and $\sigma(g_j)$. In this case, the oracle responds with the unique handle corresponding to $g_i \pm g_j$. It follows that any query of the adversary can be expressed as a linear combination of the initial handles given to the adversary. As with the random oracle model [BR93], the group oracle can be computed on demand (for the purposes of simulation) in polynomial space and time (if the adversary makes a polynomial number of queries) using a table lookup.

As a reminder we consider error distributions $\mathbf{e}'$ given by the coordinatewise product of a uniform vector $\mathbf{e} \in \mathbb{Z}_q^n$ and a "selection vector" $\mathbf{s} \in \{0,1\}^n$: that is, $e'_i = e_i \cdot s_i$. The condition we require on $\mathbf{s}$ is subset entropy (see Definition 1). With this background, we can state our main theorem:

Theorem 1. *Let λ be a security parameter. Let $q = q(\lambda)$ be a prime and $n = n(\lambda), k = k(\lambda)$ be integers with $k \leq n \leq q$. Let $\mathbf{A} \in (\mathbb{Z}_q)^{n \times k}$, $\mathbf{e}' \in (\mathbb{Z}_q)^n, \mathbf{x} \in (\mathbb{Z}_q)^k$ be uniformly distributed. Let $\ell \in \mathbb{Z}^+$ be a free parameter. Let $W \in \{0,1\}^n$ be a distribution with $(\alpha, k - (\ell - 1))$-entropy subsets. Define the vector $\mathbf{e} = \mathbf{s} \cdot_c \mathbf{e}'$, where $\mathbf{s} \leftarrow S$ and $\cdot_c$ is componentwise multiplication. Lastly, let $\mathbf{U} \in (\mathbb{Z}_q)^n$ be uniformly distributed. Then for all* generic *adversaries $\mathcal{D}$ making at most m queries*

$$\Pr[\mathcal{D}(\mathbf{A}, \mathbf{A}\mathbf{x} + \mathbf{e}) = 1] - \Pr[\mathcal{D}(\mathbf{A}, \mathbf{U}) = 1] < 2\left(\frac{m(m+1)}{2}\left(\frac{1}{2^\alpha} + \frac{2}{q}\right) + \frac{\binom{n}{k}}{q^\ell}\right).$$

In particular, if $\alpha = \omega(\log \lambda)$, $q = \omega(\mathtt{poly}(\lambda)), n = \mathtt{poly}(\lambda)$, $m = \mathtt{poly}(\lambda)$ and $\binom{n}{k}/q^\ell = \mathtt{ngl}(\lambda)$, then the statistical distance between the two cases is $\mathtt{ngl}(\lambda)$.

Proof of Theorem 1. Most generic group proofs follow the same basic structure showing that an adversary $\mathcal{D}$ cannot distinguish between two cases except when some unlikely event happens, typically associated with a linear degeneracy. This notion was formalized by Bishop et al. [BKM+18] in what they called the *simultaneous oracle indistinguishability game*. The core of our proof demonstrates oracle indistinguishability of the two distributions $\mathbf{A}, \mathbf{A}\mathbf{x} + \mathbf{e}$ and $\mathbf{A}, \mathbf{U}$ (Lemma 1). Before introducing the oracle

indistinguishability setting, we adapt the standard generic group setting in a way that simplifies the analysis and gives the adversary more power.

We will denote handles $\sigma_a(g)$ given by an oracle $\mathcal{O}_a$ as outputs of a function σ_a which—for a particular group G—maps to a sufficiently large output space $\{0,1\}^*$ so handles are independently random for each group element and hence unique with overwhelming probability. Importantly, no structure of the group is admitted through the representation. It is easy to show that an adversary in this model cannot query a group handle that was not, in fact, provided in response to a previous query (except with negligible probability). It follows that an adversary is provided a list of encodings every future query has an immediate, operational interpretation as a linear combination of the original encodings (with coefficients known to the adversary).

For our analysis, it is convenient to strengthen the ability of an algorithm interacting with the group oracle by explicitly permitting linear combination queries of elements of a noisy codeword. Specifically, we allow the adversary to make queries that represent linear combinations of $n+1$ encodings representing the group identity and the noisy codeword elements $(\sigma_a(g_0 = 0), \sigma_a(g_1), \ldots, \sigma_a(g_n))$ via a vector $\vec{\chi} = (\chi_0, \chi_1, \ldots, \chi_n)$ of $n+1$ elements of $\mathbb{Z}_q$. The oracle responds to such a query with the handle for the element $\sigma(\sum_{i=0}^{n} \chi_i \cdot g_i)$. We call an adversary with this query structure *generic*. This is an increase in power; however, such an arbitrary linear combination query can be simulated by a standard adversary (permitted queries with two known handles at a time) with polynomial overhead using repeated squaring.

The Two Oracle Distinguishability Game. As stated above the basic structure of a generic proof is showing that responses between two oracles $\mathcal{O}_1$ and $\mathcal{O}_2$ (in our case a noisy random linear code and a truly uniform value) cannot be distinguished unless some bad event happens. Bishop et al. [BKM+18, Section 4] introduce the simultaneous oracle game as a technical tool for showing indistinguishability. In this model, the adversary is given two oracles $\mathcal{O}_1$ and a second oracle $\mathcal{O}^*$ that is either $\mathcal{O}_1$ or $\mathcal{O}_2$ with probability 1/2. Since both oracles are generating handles uniformly at random, the adversary's only distinguishing ability can come from detection of repetition in handles.

Specifically, the adversary $\mathcal{D}$ builds a table ζ and two functions Ψ and Ψ^{-1} where ζ has two columns to store handles for each oracle. This table naturally induces Ψ, Ψ^{-1} where Ψ is a function from handles for the $\mathcal{O}_1$ to the handles for $\mathcal{O}_2$, Ψ^{-1} takes handles from $\mathcal{O}_2$ to the handles from $\mathcal{O}_1$. The table and the two functions are constructed on the fly as $\mathcal{D}$ makes queries. Ψ and Ψ^{-1} are one-to-one functions up until a collision, in which case $\mathcal{D}$ stops and is treated as succeeding in distinguishing. We treat them as one-to-one functions for our analysis.

We say a handle is *known* if the adversary has received that handle for the same oracle and stored it in ζ. A handle is *new* when it is the first time the adversary has been given that handle by that oracle.

When receiving handles h_1, h_2 for $\mathcal{O}_1$ and $\mathcal{O}_2$ respectively, the adversary looks up each handle in the appropriate column of ζ and categorizes the two handles as known or new based on the definitions above.

1. If one handle is new and the other is known, then we say the handles are inconsistent and the adversary stops.

2. If both handles are known, the adversary checks that $\Psi(h_1) = h_2$ and $\Psi^{-1}(h_2) = h_1$. If both are true, then the handle are said to be consistent. If either is not the case, the handles are also inconsistent and the adversary stops.

3. If both handles are new, the adversary adds both to the appropriate column of ζ, sets $\Psi(h_1) = h_2$ and $\Psi^{-1}(h_2) = h_1$.

To summarize, we say a query response (a pair of handles) from two oracles is *consistent* when the responses are either both new, or both are known. The adversary can only distinguish the oracles apart when it receives inconsistent handles in response to a query.

In our analysis, the security game keeps track of the linear combinations of the queries that the adversary makes to the oracles. Our reduction uses two oracles, one of which uses $\mathbf{Ax}+\mathbf{e}$ to generate the noisy codeword. We call this oracle the code oracle or $\mathcal{O}_c$. The other oracle responds with new handles for each new query. We call this oracle the random oracle or $\mathcal{O}_r$. We stress that in both cases the adversary and both oracles are given access to the actual generating matrix $\mathbf{A}$. Notably, the adversary can use the generating matrix to create its queries.

Code Oracle. We define a code oracle that responds to queries faithfully. We denote this oracle $\mathcal{O}_c$.

This oracle picks a message $\mathbf{x}$, uses the generating matrix $\mathbf{A}$, samples an *error selection vector* $\mathbf{s} \in \{0,1\}^n$ according to the distribution S with $(\alpha, k-(l-1))$-entropy subsets, and a random vector $\mathbf{e} \in \mathbb{Z}_q^n$.

With probability $\binom{n}{k}/q^l$, there will exist a $k \times k$ minor of $\mathbf{A}$ will not have rank $k-l$. As we will see this means that the adversary may create a query in the null space of the code that depends on at most $k-l-1$ positions. This means that there is no guarantee on the entropy of the adversarially selected subset. This leads to an additional failure probability $\binom{n}{k}/q^l$. We count this probability of degeneracy in our lemma later. We note this probability is entirely dependent on the sampling of $\mathbf{A}$.

The oracle begins by calculating the noisy codeword $\mathbf{b}$ as follows:

$$\mathbf{b} = \mathbf{Ax} + \mathbf{e}' \cdot_c \mathbf{s}.$$

When queried with a vector $\vec{\chi} = (\chi_0, \chi_1, \ldots, \chi_n) \in \mathbb{Z}_q^n$ the oracle answers with an encoded group element $\sigma_c(\sum_{i=0}^n \chi_i \cdot b_i)$.

Random Oracle. We also define an oracle $\mathcal{O}_r$ that creates n random initial encodings and responds to all requests for linear combinations with distinct random elements. For a sequence of indeterminates $\mathbf{y} = (y_0, y_1, \ldots, y_n)$, this oracle can be described as a table where the left side is a vector representing a linear combination of the indeterminates and the right side is a handle associated with each vector.

When presented a query, if the vector is in the oracles table, it responds with the handle on the right side of the table. When the query is a new linear combination, it generates a distinct handle. The adversary then stores the vector and the handle in the table and sends the handle to $\mathcal{D}$. We denote the handles τ_i to distinguish them from the encoded group elements of the code oracle.

Lemma 1. *In the simultaneous oracle model, the probability that some adversary $\mathcal{D}$, when interacting with two group oracles $\mathcal{O}_c$ and $\mathcal{O}^*$ (which is either $\mathcal{O}_c$ or $\mathcal{O}_r$ with probability $1/2$) differs after m queries by at most*

$$\left|\Pr[\mathcal{D}^{\mathcal{O}_c}(\cdot)=1] - \Pr[\mathcal{D}^{\mathcal{O}^*}(\cdot)=1]\right| \leq \frac{m(m+1)}{2}\left(\frac{1}{2^\alpha}+\frac{1}{q}\right) + \frac{\binom{n}{k}}{q^l}.$$

Proof. We examine the simultaneous oracle game that the adversary plays between $\mathcal{O}_c$ and $\mathcal{O}^*$. The adversary builds its table ζ and functions Ψ and Ψ^{-1} as described above. We present a table for after the adversary queries the generator and the noisy codeword elements to illustrate the functionality of ζ, Ψ, and Ψ^{-1}. For clarity we denote handles returned by the first oracle using $\sigma_{c,1}$. We also provide the query for each set of handles. In the case where $\mathcal{O}^* = \mathcal{O}_c$ we denote the second encoding function $\sigma_{c,2}$ and the second oracle $\mathcal{O}_{c,2}$. Similarly, we refer to the first code oracle as $\mathcal{O}_{c,1}$. The only common randomness between these encodings is $\mathbf{A}$ which is also known by $\mathcal{D}$.

Linear Combination	$\mathcal{O}_{c,1}$ handle	$\mathcal{O}_{c,2}$ handle -OR-	$\mathcal{O}_r$ handle
$(1, 0, \ldots, 0)$	$\sigma_{c,1}(0)$	$\sigma_{c,2}(0)$	τ_0
$(0, 1, \ldots, 0)$	$\sigma_{c,1}(b_1)$	$\sigma_{c,2}(b'_1)$	τ_1
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$(0, 0, \ldots, 1)$	$\sigma_{c,1}(b_n)$	$\sigma_{c,2}(b'_n)$	τ_n

In addition to this information we also analyze the underlying structure of $\mathcal{O}_{c,1}$ (noting that $\mathcal{O}_{c,2}$ retains the same underlying structure). We do this by noticing that the group element b_i is $\sigma(\mathbf{A}_i\mathbf{x} + \mathbf{e}_i \cdot \mathbf{s}_i)$ (we use $\mathbf{A}_i$ to denote the ith row of a matrix $\mathbf{A}$). We denote the Diagonal matrix induced by a vector $\mathbf{y}$ as $Diag(\mathbf{y})$.

$$\sum_{i=0}^{n} \chi_i b_i = \sum_{i=0}^{n} \chi_i(\mathbf{A}_i \cdot \mathbf{x}) + \sum_{i=0}^{n} \chi_i(Diag(\mathbf{s})_i \cdot \mathbf{e})$$

$$= \sum_{i=0}^{n} \chi_i \cdot \left(\left[\begin{array}{cccc|cccc} 1 & 0 & \ldots & 0 & s_1 & 0 & \ldots & 0 \\ 0 & 1 & \ldots & 0 & 0 & s_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & 1 & 0 & 0 & \ldots & s_n \end{array}\right]_i \cdot \begin{bmatrix} c_1 \\ \vdots \\ c_n \\ e_1 \\ \vdots \\ e_n \end{bmatrix} \right) \qquad \text{(where } c_j \text{ is } \mathbf{A}_j \cdot \mathbf{x}\text{)}$$

$$= \sum_{i=0}^{n} \chi_i \cdot \left(\left[\begin{array}{cccc|cccc} A_{0,0} & A_{0,1} & \ldots & A_{0,k} & s_1 & 0 & \ldots & 0 \\ A_{1,0} & A_{1,1} & \ldots & A_{1,k} & 0 & s_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ A_{n,0} & A_{n,1} & \ldots & A_{n,k} & 0 & 0 & \ldots & s_n \end{array}\right]_i \cdot \begin{bmatrix} x_1 \\ \vdots \\ x_k \\ e_1 \\ \vdots \\ e_n \end{bmatrix} \right)$$

Using this latter representation, note that linear combinations of the query vectors χ also apply to the error selection vector $\mathbf{s}$ meaning that if the query vector has Hamming weight greater than k there will be k error selection components w_i in the resulting group element's handle. Similarly, if two queries $\vec{\chi}_i$ and $\vec{\chi}_j$ differ in at least k positions then the induced error selection vector $\vec{\chi}' = (\vec{\chi}_i - \vec{\chi}_j)$ contains at least k nonzero components.

Alternatively, $\mathcal{O}_r$ responds to each distinct query with a new handle. This means that there is exactly one occasion to distinguish when $\mathcal{O}^* = \mathcal{O}_{c,2}$ or $\mathcal{O}^* = \mathcal{O}_r$. This is when the handle returned by $\mathcal{O}_{c,1}$ is known and τ_i is new. We separate the analysis into two cases: group collisions due to noise and null space query differences.

Group Collisions Due to Error. $\mathcal{O}_c$ carries out calculations with actual group elements. Thus, when a group element is generated, there is a small probability that the linear combination produces a group element whose handle is known to the adversary. This happens for each query with probability equal to the number of known handles over the size of the group because errors are uniform in the field; in particular, linear operations of any non-zero number of error terms will also be uniformly distributed.

Since each query can generate at most one new handle we upper bound the probability by summing the probability at each step over m queries as $\sum_{i=1}^{m}(i/q) = (1/q) \cdot (m(m+1)/2)$.

Null Space Query Differences. Conditioned on every $k \times k$ minor of the generating matrix of the code having rank at least $k - l$ (as mentioned above this happens with probability at least $1 - \binom{n}{k}/q^l$), when the difference $\vec{\chi}' = \vec{\chi}_i - \vec{\chi}_j$ between two queries is in the null space of the code, that is when $\vec{\chi}'\mathbf{A} = 0$ then the number of nonzero components in $\vec{\chi}'$ is at least $k - l$. This means that there are at least $k - l$ error components in the subsets of $\mathbf{s}$ induced by $\vec{\chi}'$. By assumption on the distribution E, then probability that this product $\vec{\chi}' \cdot_c \mathbf{s}$ is equal to any particular value is at most $2^{-\alpha}$. If the induced value $\vec{\chi}' \cdot_c \mathbf{s}$ is a previously known group element then there is a collision. This probability for each query is the number of known handles at the time of the query over the size of the field. After m queries, we take the sum of the probability for each query and offer an upper bound $\sum_{i=1}^{M}(i/q) \cdot 2^{-\alpha} = (m(m+1)/2) \cdot (1/2^{\alpha})$. Otherwise, the query has an error term and collides with probability given in the first case. We take a union bound over the two cases after m queries for the lemma's result. □

This lemma gives us the distinguishing power of an adversary interacting with our code oracle and our random oracle. Our random oracle never has collisions because it creates fresh handles every time. To create an oracle analogous to a uniform distribution as claimed in Theorem 1 we introduce another term for random failure with the same probability as group collisions due to error. We call this last oracle the uniform oracle and use it in our analysis with the added probability of failure.

Taking the result of this technical lemma, we can prove Theorem 1 using the contrapositive of [BKM+18, Lemma 7] given here for completeness.

Lemma 2 ([BKM+18] Lemma 7)**.** *Suppose there exists an algorithm $\mathcal{A}$ such that*

$$|\Pr[\mathcal{A}^{\mathcal{G}_{\mathcal{M}}}(\mathcal{O}^{\mathcal{G}_{\mathcal{M}}}) = 1] - \Pr[\mathcal{A}^{\mathcal{G}_{\mathcal{S}}}(\mathcal{O}^{\mathcal{G}_{\mathcal{S}}}) = 1]| > \delta.$$

Then an adversary can win the simultaneous oracle game with probability at least $\frac{1}{2} + \frac{\delta}{2}$ for any pair of oracles $(\mathcal{G}_{\mathcal{M}}, \mathcal{G}_* = \mathcal{G}_{\mathcal{M}}/\mathcal{G}_{\mathcal{S}})$.

In our scheme, we use the contrapositive and use our result from Lemma 1 (and the modification to the uniform oracle) where

$$\delta/2 = \frac{m(m+1)}{2}\left(\frac{1}{2^{\alpha}} + \frac{2}{q}\right) + \frac{\binom{n}{k}}{q^l}.$$

Since the probability of an adversary winning the simultaneous oracle game is bounded above by

$$1/2 + \frac{m(m+1)}{2}\left(\frac{1}{2^{\alpha}} + \frac{2}{q}\right) + \frac{\binom{n}{k}}{q^l}$$

then

$$\Pr[A(\mathcal{O}_c) = 1] - \Pr[A(\mathcal{O}_r) = 1] < 2\left(\frac{m(m+1)}{2}\left(\frac{1}{2^{\alpha}} + \frac{2}{q}\right) + \frac{\binom{n}{k}}{q^{\ell}}\right).$$

Because $\mathcal{O}_r$ represents the oracle for uniform randomness and $\mathcal{O}_c$ is the oracle for $A\mathbf{x} + \mathbf{e}$, this gives us the result for generic adversaries. □

4 Application to Fuzzy Extractors - Code Offset in the Exponent

Our primary application is a new fuzzy extractor that performs error correction "in the exponent." A fuzzy extractor is a pair of algorithms designed to extract stable keys from a physical randomness source that has entropy but is noisy. If repeated readings are taken from the source one expects these readings to be close according to a distance metric but not identical. Before introducing the construction we review the definition. We consider a generic group version of security (computational security is defined in [FMR13], information-theoretic security in [DORS08]).

Definition 2. *Let $\mathcal{W}$ be a family of probability distributions over $\mathcal{M}$. A pair of procedures* $(\mathsf{Gen} : \mathcal{M} \to \{0,1\}^\kappa \times \{0,1\}^*, \mathsf{Rep} : \mathcal{M} \times \{0,1\}^* \to \{0,1\}^\kappa)$ *is an* $(\mathcal{M}, \mathcal{W}, \kappa, t)$-computational fuzzy extractor *that is* (ϵ_{sec}, m)*-hard with error* δ *if* Gen *and* Rep *satisfy the following properties:*

- Correctness*: if* $\mathsf{dis}(\mathbf{w}, \mathbf{w}') \le t$ *and* $(\mathsf{key}, \mathsf{pub}) \leftarrow \mathsf{Gen}(\mathbf{w})$*, then* $\Pr[\mathsf{Rep}(\mathbf{w}', \mathsf{pub}) = r] \ge 1 - \delta$.
- Security*: for any distribution* $W \in \mathcal{W}$*, the string* key *is close to random conditioned on* pub *for all generic* $\mathcal{A}$ *making at most* m *queries to the group oracle* $\mathcal{O}$*, that is*

$$\Pr[\mathcal{A}^{\mathcal{O}}(\mathsf{Key}, P) = 1] - \Pr[\mathcal{A}^{\mathcal{O}}(U, P) = 1] \le \epsilon_{sec}.$$

In the above definition, the errors are chosen before P: if the error pattern between $\mathbf{w}$ and $\mathbf{w}'$ depends on the output of Gen, then there is no guarantee about the probability of correctness.

4.1 Construction

The code-offset construction is a conceptually simple fuzzy extractor [DORS08]. The idea is for p to be a one time pad of w.[4] That is, $p = c \oplus w$. The Rep algorithm has p and w' as inputs and computes $c' = p \oplus w'$. Importantly, if $\mathsf{dis}(w, w') \le t$ then $\mathsf{dis}(c, c') \le t$. If c is chosen from a suitable error correcting code, then it is possible to decode to c and recover w.

Importantly, if c is chosen from a error correcting code, it cannot be a uniform point and thus the one-time pad analysis does not apply. However, the conditional entropy of W given P or $\tilde{\mathrm{H}}_\infty(W \mid P)$ reduces by at most the gap between the size of the uniform distribution and the error correction code. That is, for code $C \in \{0,1\}^n$:

$$\tilde{\mathrm{H}}_\infty(W \mid P) \ge \mathrm{H}_\infty(W) - (n - \log|C|).$$

The major problem with the code offset construction is the limited applicability to physical distributions W. In particular, for many distributions W this analysis provides no guarantee on the strength of the derived key (see discussion in [CFP+16]). To address this problem, Fuller et al. [FMR13] proposed to replace a structured code with a random linear code and rely on the hardness of LWE. Subsequent work adapted the construction to a binary field [HRvD+16] and showed how to make the construction a reusable fuzzy extractor [ACEK17].

These constructions are a code-offset construction with a random code: $\mathsf{pub} = (\mathbf{A}, \mathbf{Ax} + \mathbf{w})$ where $\mathbf{A}$ and $\mathbf{x}$ are random. For a suitable $\mathbf{w}$ the value pub is pseudorandom. The associated decoding procedure is a simple guess and check, finding subsets $\mathcal{I}$ and then computing $\mathbf{x} = \mathbf{A}_{\mathcal{I}}^{-1}\mathsf{pub}_{\mathcal{I}} - \mathbf{w}'_{\mathcal{I}} = \mathbf{x}\mathbf{A}_{\mathcal{I}}^{-1}(\mathbf{w} - \mathbf{w}')_{\mathcal{I}}$. To achieve a hard LWE instance their correction capability was only $t = \Theta(\log|\mathbf{w}|)$ which is inadequate

[4] The cryptographic key is produced by applying a randomness extractor on w. [NZ93].

for most physical sources. Achieving a higher error tolerance could be achieved by reducing the dimension of the code $|\mathbf{x}|$ but this has a direct effect on the hardness of the underlying lattice problem.

To address this issue we move the LWE code-offset to the exponent. The construction is:

Construction 1. *Let λ be a security parameter and let $k = k(\lambda)$ be some parameter where $k = \omega(\log \lambda)$. Let α be a free parameter. Let $p = p(\lambda)$ be an ensemble of primes. Let $\mathbb{Z}_p$ be the field with p elements. Let $\mathcal{W} \in \{0,1\}^n$ be a metric space and let $k = \omega(\log \lambda)$ be a parameter. Let $\tau = \max(0.01, t/n)$. Define $(\mathsf{Gen}, \mathsf{Rep})$ as follows:*

Gen

1. ***Input:*** $w = w_1, ..., w_n$
2. *Sample random generator r of $\mathbb{Z}_p^*$.*
3. *Sample $\mathbf{A} \leftarrow (\mathbb{Z}_p)^{n \times (k+\alpha)}, \mathbf{x} \leftarrow (\mathbb{Z}_p)^{k+\alpha}$.*
4. *Sample $\mathbf{y} \xleftarrow{\$} \{0,1\}^n$.*
5. *For $i = 1, ..., n$:*
   - *(i) If $w_i = y_i$, set $\mathsf{c}_i = r^{\mathbf{A}_i \cdot x}$.*
   - *(ii) Else set $\mathsf{c}_i \xleftarrow{\$} \mathbb{Z}_p^*$.*
6. *Set $\mathsf{key} = r^{\mathbf{x}_{0...\alpha-1}}$.*
7. *Output (key, p), $p = (r, \mathbf{y}, \mathbf{A}, \{\mathsf{c}_i\}_{i=1}^n)$.*

Rep

1. ***Input:*** $(w', p = (r, \mathbf{y}, \mathbf{A}, \mathsf{c}_1 \dots \mathsf{c}_\ell))$
2. *Let $\mathcal{I} = \{i | w'_i = y_i\}$.*
3. *For $i = 1, ..., \ell$:*
   - *(i) Choose random $J_i \subseteq \mathcal{I}$ where $|J| = k$.*
   - *(ii) If $\mathbf{A}_{J_i}^{-1}$ does not exist go to 4.*
   - *(iii) Compute $\mathsf{c}' = r^{\mathbf{A}(\mathbf{A}_{J_i}^{-1} \mathbf{c}_{J_i})}$.*
   - *(iv) If $\mathsf{dis}(\mathsf{c}_{\mathcal{I}}, \mathsf{c}'_{\mathcal{I}}) \leq |\mathsf{c}_{\mathcal{I}}|(1 - 2\tau)$, output $\mathsf{key} = r_{0...\alpha-1}^{\mathbf{A}_{J_i}^{-1} \mathbf{c}_{J_i}}$.*
4. *Output $\perp$.*

Theorem 2. *Let all parameters be as in Construction 1. Let $\ell \in \mathbb{Z}^+$ be a free parameter. Let $W \in \{0,1\}^n$ be a distribution with $(\alpha, k - (\ell - 1))$-entropy subsets. define the random variable*

$$\mathbf{E} \stackrel{def}{=} \begin{cases} U_{\mathbb{Z}_p^*} & \text{if } Y_i \neq W_i, \\ 0 & \text{if } Y_i = W_i. \end{cases}$$

Then for all generic adversaries $\mathcal{D}$ making at most m queries to $\mathcal{O}$ then

$$\Pr[\mathcal{D}(\mathsf{Key}, R, Y, \mathbf{A}, \mathbf{Ax} + \mathbf{E}) = 1] - \Pr[\mathcal{D}(U_{\mathbb{Z}_p^*}^{\alpha}, R, Y, \mathbf{A}, \mathbf{Ax} + E) = 1]$$
$$\leq 2 \left(\frac{m(m+1)}{2} \left(\frac{1}{2^\alpha} + \frac{2}{q} \right) + \frac{\binom{n}{k}}{q^\ell} \right).$$

That is for generic adversaries making m queries, Construction 1 is a secure fuzzy extractor if $\alpha = \omega(\log \lambda), q = \omega(\mathtt{poly}(\lambda)), m = \mathtt{poly}(\lambda), \binom{n}{k}/q^\ell = \mathtt{ngl}(\lambda)$ for some $\epsilon_{sec} = \mathtt{ngl}(\lambda)$.

Proof. The main technical content of our proof is in Theorem 1 which shows hardness of the decisional version of LWE for error distributions with subset entropy for generic adversaries. To achieve the above theorem we need two additional notes:

1. A lemma by Akavia, Goldwasser, and Vaikuntanathan [AGV09, Lemma 2] which states that if the decision version of LWE is hard for k dimensions than any additional α dimensions are hardcore. To apply this Lemma in our setting we simply have to note that the reduction works for a generic adversary, it only needs to sample random group elements and compute linear functions.

2. Instead of directly using W as an error selection vector we use when $Y_i \neq W_i$ as the error selection vector. This change is made only for simplifying correctness analysis. In particular, since W and Y are independent, define the random variable S where $S_i = (Y_i \stackrel{?}{=} W_i)$. Then since Y is uniformly distributed when W has $(\alpha, k-(\ell-1))$-entropy subsets so does S. The error selection vector is defined by S.

This completes the proof of Theorem 2. □

Correctness and Efficiency We now show the construction is also correct and efficient. Our correctness argument considers constant $k = \Theta(n)$ and $t = \Theta(n)$. For the fuzzy extractor application, one would consider a smaller k and t. In particular, for $t = o(n)$ the theorem applies with overwhelming probability as long as $k \leq \frac{(1-\Theta(1))}{3} * n$. We use the q-ary entropy function which is a generalization of the binary entropy function to larger fields. $H_q(x)$ is the q-ary entropy function defined as

$$H_q(x) = x \log_q(q-1) - x \log_q(x) - (1-x) \log_q(1-x).$$

Theorem 3. *Let parameters be as in Construction 1. Define $\tau = t/n$. Let $0 < \delta < 1 - H_q(4\tau)$ and suppose that $k \leq (1/3) \cdot \lceil 1 - H_q(4\tau) - \delta \rceil n$. If* Rep *outputs a value other than $\perp$ it is correct with probability at least $1 - e^{-\Theta(n)}$.*

of Theorem 3. We assume a fixed number of iterations in Rep denoted by ℓ. Recall we assume that $\mathsf{dis}(w, w') \leq t$ and that the value $\mathbf{y}$ is independent of both values (by Def 2, w' does not depend on the public value). We first consider the final check of whether $\mathsf{dis}(\mathsf{c}_{\mathcal{I}}, \mathsf{c}'_{\mathcal{I}}) \leq |\mathsf{c}_{\mathcal{I}}|(1-2\tau)$ will return correctly if and only $r^{\mathbf{x}} = r^{\mathbf{A}_{J_i}^{-1} \mathsf{c}_{J_i}}$. We stress that this is property is independent of the chosen subset and only depends on $\mathbf{A}, \mathbf{x}, w, w'$ and y. To simplify notation we refer to the values in the exponent, but our argument directly applies to the generated group elements.

Define the matrix $\mathbf{A}_{\mathcal{I}}$ defined by the set $\mathcal{I}$. By Chernoff bound,

$$\Pr\left[|\mathcal{I}| \leq \left(1 - \frac{1}{3}\right) \mathbb{E}\,|\mathcal{I}|\right] = \Pr\left[|\mathcal{I}| \leq \left(\frac{2}{3}\right)\frac{n}{2}\right] \leq e^{-\frac{n}{36}} \leq e^{-\Theta(n)}.$$

Without loss of generality we assume that the size of $\mathcal{I} = n/3$.

Consider some fixed w, w' such that $\mathsf{dis}(w, w') \leq t$ and define the random variable Z of length n where a bit i of z that indicates when $w_i = w'_i$ and when $w'_i = y_i$. We consider the setting when $t = \Theta(n)$, if $t = o(n)$ then $\tau = .01$ and the condition holds with high probability.

We can lower bound of weight of S by a binomial distribution with $n - t$ flips and probability $1/2$. That is, $\mathbb{E}[S] = |\mathcal{I}| * (1-\tau) \geq (n/3)(1-\tau)$. By an additive Chernoff bound,

$$\Pr\left[S - \mathbb{E}[S] \geq t\right] = \Pr\left[S - \mathbb{E}[S] \geq \tau n\right] \leq 2e^{-2\tau^2 n} \leq e^{-\Theta(n)}.$$

To show correctness it remains to show that $\mathbf{x}$ is unique. We again assume that $\mathcal{I} = n/3$, all arguments proceed similarly when $\mathcal{I} > n/3$. To show uniqueness of $\mathbf{x}$ suppose that there exists two $\mathbf{x}_1, \mathbf{x}_2$ such that $\mathsf{dis}(\mathbf{A}_{\mathcal{I}}\mathbf{x}_1, \mathsf{c}_{\mathcal{I}}) \leq |\mathsf{c}_{\mathcal{I}}|(1-2\tau)$ and $\mathsf{dis}(\mathbf{A}_{\mathcal{I}}\mathbf{x}_2, \mathsf{c}_{\mathcal{I}}) \leq |\mathsf{c}_{\mathcal{I}}|(1-2\tau)$. This means that $\mathbf{A}_{\mathcal{I}}(\mathbf{x}_1 - \mathbf{x}_2)$ contains at most $4t/3$ nonzero components. To complete the proof we use the following standard theorem:

Lemma 3. *[Gur10, Theorem 8] For prime $q, \delta \in [0, 1-1/q), 0 < \epsilon < 1 - H_q(\delta)$ and sufficiently large n, the following holds for $k = \lceil (1 - H_q(\delta) - \epsilon)n \rceil$. If $\mathbf{A} \in \mathbb{Z}_q^{n \times k}$ is drawn uniformly at random, then the linear code with $\mathbf{A}$ as a generator matrix has rate at least $(1 - H_q(\delta) - \epsilon)$ and relative distance at least δ with probability at least $1 - e^{-\Omega(n)}$.*

Application of Lemma 3 completes the proof of Theorem 3. □

Recovery Our analysis of running time is similar in spirit to that of Canetti et al. [CFP+16]. For any given i, the probability that $w'_{\mathcal{J}_i} = w_{\mathcal{J}_i}$ is at least

$$\left(1 - \frac{2t}{n-k}\right)^k.$$

Therefore, the probability that no iteration matches is at most

$$\left(1 - \left(1 - \frac{2t}{n-k}\right)^k\right)^\ell.$$

We can use the approximation $e^x \approx 1 + x$ to get

$$\left(1 - \left(1 - \frac{2t}{n-k}\right)^k\right)^\ell \approx (1 - e^{-\frac{2tk}{n-k}})^\ell \approx \exp(-\ell e^{-\frac{2tk}{n-k}}).$$

Suppose that correctness $1 - \delta \geq 1 - (\delta' + e^{-\Theta(n)})$ is desired. (Here, the $e^{-\Theta(n)}$ term is due to sampling of a bad matrix $\mathbf{A}$ and failures of Chernoff bounds above.) Then if $k = o(n)$ with $tk = cn \ln n$ for some constant c, setting $\ell \approx n^{2c+\Theta(1)} \log \frac{1}{\delta'}$ suffices as:

$$\begin{aligned}
\exp\left(-\ell e^{-\frac{tk}{n-k}}\right) &= \exp\left(-n^{2c} \log \frac{1}{\delta'} e^{-\frac{2tk}{n-k}}\right) \\
&\leq \exp\left(-n^{2c+\Theta(1)} * \log \frac{1}{\delta'} * e^{-(2c+o(1)) \ln n}\right) \\
&= \exp\left(-n^{2c+\Theta(1)} * \log \frac{1}{\delta'} * n^{-(2c+o(1))}\right) \\
&\leq \delta'
\end{aligned}$$

In particular, in the generic group model for k that is only slightly larger than $\omega(\ln n)$ one can support error rates $t = o(n)$ only slightly smaller than constant rates.

4.2 Reusability

Reusability is the ability to support multiple independent enrollments of the same value, allowing users to reuse the same biometric or PUF, for example, with multiple noncooperating providers. More precisely, the algorithm Gen may be run multiple times on correlated readings $w^1, ..., w^\rho$ of a given source. Each time, Gen will produce a different pair of values $(r^1, p^1), ..., (r^\rho, p^\rho)$. Security for each extracted string r^i should hold even in the presence of all the helper strings $p^1, \ldots, p^\rho$ (the reproduction procedure Rep at the ith provider still obtains only a single w' close to w^i and uses a single helper string p_i). Because providers may not trust each other each r_i should be secure even when all r_j for $j \neq i$ are also given to the adversary.

Definition 3 (Reusable Fuzzy Extractor [CFP+16])**.** *Let $\mathcal{W}$ be a family of distributions over $\mathcal{M}$. Let $(\mathsf{Gen}, \mathsf{Rep})$ be a $(\mathcal{M}, \mathcal{W}, \kappa, t)$-computational fuzzy extractor that is (ϵ_{sec}, m)-hard with error δ. Let $(W^1, W^2, \ldots, W^\rho)$ be ρ correlated random variables such that each $W^j \in \mathcal{W}$. Let D be an adversary. Define the following game for all $j = 1, ..., \rho$:*

- ***Sampling*** *The challenger samples* $w^j \leftarrow W^j$ *and* $u \leftarrow \{0,1\}^\kappa$.
- ***Generation*** *The challenger computes* $(r^j, p^j) \leftarrow \mathsf{Gen}(w^j)$.
- ***Distinguishing*** *The advantage of* D *is*

$$\begin{aligned} Adv(D) \stackrel{def}{=} &\Pr[D(r^1, ..., r^{j-1}, r^j, r^{j+1}, ..., r^\rho, p^1, ..., p^\rho) = 1] \\ &- \Pr[D(r^1, ..., r^{j-1}, u, r^{j+1}, ..., r^\rho, p^1, ..., p^\rho) = 1]. \end{aligned}$$

$(\mathsf{Gen}, \mathsf{Rep})$ *is* $(\rho, \epsilon_{sec}, m)$*-reusable if for all generic* D *making at most* m *queries and all* $j = 1, ..., \rho$*, the advantage is at most* ϵ_{sec}.

As mentioned in the introduction, Construction 1 is also a reusable fuzzy extractor against generic adversaries.

Theorem 4. *Let all parameters be as in Construction 1. Let* $\ell \in \mathbb{Z}^+$ *be a free parameter. Let* $W \in \{0,1\}^n$ *be a distribution with* $(\alpha, k - (\ell - 1))$*-entropy subsets. define the random variable*

$$\mathbf{E} \stackrel{def}{=} \begin{cases} U_{\mathbb{Z}_p^*} & \text{if } Y_i \neq W_i, \\ 0 & \text{if } Y_i = W_i. \end{cases}$$

Then $(\mathsf{Gen}, \mathsf{Rep})$ *is a* $(\rho, \epsilon_{sec}, m)$ *reusable fuzzy for all generic adversaries making at most* m *queries where*

$$\epsilon_{sec} = 2\rho \left(\frac{m(m+1)}{2} \left(\frac{1}{2^\alpha} + \frac{2}{q} \right) + \frac{\binom{n}{k}}{q^\ell} \right).$$

For generic adversaries making m *queries, Construction 1 is a reusable secure fuzzy extractor if* $\alpha = \omega(\log \lambda), q = \omega(\mathtt{poly}(\lambda)), m = \mathtt{poly}(\lambda)$, $\rho = \mathtt{poly}(\lambda)$, $\binom{n}{k}/q^\ell = \mathtt{ngl}(\lambda)$ *for some* $\epsilon_{sec} = \mathtt{ngl}(\lambda)$.

Proof. This argument requires a little understanding of the generic group proof from Section 3. This argument showed that an adversary knowing $\mathbf{A}$ was unable to distinguish between $\mathbf{Ax} + \mathbf{e} \cdot_c \mathbf{s}$ from $\mathbf{U}$ except with negligible probability. Without loss of generality, we assume that the adversary is trying to learn information about the first key. For the construction to be reusable for all distinguishers, it must be true that:

$$\begin{aligned} &|\Pr[\mathcal{D}(\mathbf{U}, r_1, \mathbf{y}_1, \mathbf{A}_1, \mathbf{A}_1\mathbf{x}_1 + \mathbf{e}_1 \cdot_c \mathbf{s}_1, \{\mathsf{key}_i, p_i\}_{i=2}^n) = 1] \\ &\quad - \Pr[\mathcal{D}(r^{\mathbf{x}_{0..\alpha-1}}, r_1, \mathbf{y}_1, \mathbf{A}_1, \mathbf{A}_1\mathbf{x}_1 + \mathbf{e}_1 \cdot_c \mathbf{s}_1, \{\mathsf{key}_i, p_i\}_{i=2}^n) = 1| \leq \epsilon_{sec}. \end{aligned}$$

Crucially, in Theorem 1, we assume that handles are in a sufficient sparse space such that handles from one oracle never represent a valid handle for oracle. Rather than initializing a joint oracle to answer all queries, one can separately initialize oracles for each application of the fuzzy extractor. This is because each application of the fuzzy extractor works for a different group generator. Crucially, one this conceptual change is made one exclude queries from the adversary with handles that involve more than one oracle. Then the $n - 1$ oracles corresponding to other enrollments w_i are the same in both settings. Using a simple hybrid argument on Theorem 1 we can replace these oracles with uniform values. Once replaced by uniform values these oracles provide no information to the adversary. The theorem follows by a final application of Theorem 1. □

4.3 Comparison with sample-then-lock

As mentioned in the introduction, Canetti et al. [CFP+16] proposed a reusable fuzzy extractor based on digital lockers called *sample-then-lock*. Intuitively, a digital locker is a symmetric encryption that is semantically secure even when instantiated with keys that are correlated and only have entropy [CKVW10]. At a high level, their construction took multiple samples $w_{\mathcal{I}_j}$ from the input biometric and use these as keys for different digital lockers, all of which contained the same key. The main drawback of this construction is that it required the construction to write down each subset to be tested. Alternatively, Construction 1 allows testing of any subset. This change is significant for two reasons:

1. Only running time grows with the number of errors, storage grows only with the size of $|w|$.
2. Many physical sources are sampled along with correlated side information that is called *confidence*. Confidence information is a secondary probability distribution Z (correlated with the reading W) that can predict the error rate in a bit W_i. When Z is large this means a bit of W is less likely to differ. Examples include the magnitude of a convolution in the iris [SSF18] and the magnitude of the difference between two circuit delays in ring oscillator PUFs [HRvD+16]. Herder et al. report that by considering bits with high confidence it is possible to reduce the effective error rate from $\tau = .10$ to $\tau = 3 \times 10^{-6}$. This confidence information could not be used in Canetti et al's work to guide subset selection as it is correlated with W. On the other hand, Construction 1 only uses this information at reproduction time so it is stored public information is not correlated with the confidence information.

On the other hand the fact that all subsets are available to an adversary does provide them with additional power. Canetti et al. were able to show security for all distributions where sampling produced entropy:

Definition 4 ([CFP+16] Sources with High Entropy Samples)**.** *Let the source $W = W_1, \ldots, W_n$ consist of strings of length n over some arbitrary alphabet $\mathcal{Z}$. For some parameters k, α we say that the source W is a source with a (α, k)-**entropy-samples** if $\tilde{H}_\infty(W_{j_1}, \ldots, W_{j_k} \mid j_1, \ldots, j_k) \geq \alpha$ for uniformly random $1 \leq j_1, \ldots, j_k \leq n$.*

Our modification to this definition in Definition 1 is the natural one. Instead of $j_1, \ldots, j_k$ being a uniform subset, it can be any subset of n.

5 Application to Pattern Matching Obfuscation

In this section we introduce a second application for our main theorem. This application is known as pattern matching obfuscation. The goal is to obfuscate a string v of length n which consists of $0, 1, \perp$ where $\perp$ is a wildcard. The obfuscated program on input $x \in \{0,1\}^n$ should output 1 if and only if $\forall i, x_i = v_i \vee v_i = \perp$. Roughly, the wildcard positions are matched automatically. We directly use definitions and the construction from the recent work of Bishop et al. [BKM+18]. Our improvement is in analysis, showing security for more distributions V. We start by introducing a definition of security:

Definition 5. *Let $\mathcal{C} = \mathcal{C}_n$ be a family of circuits where $\mathcal{C}_n$ takes inputs of length n and let $\mathcal{O}$ be a PPT algorithm taking $n \in \mathbb{N}$ and $C \in \mathcal{C}$ outputting a new circuit C'. Let $\mathcal{D} = \mathcal{D}_n$ be an ensemble of distribution families where each $D \in \mathcal{D}_n$ is a distribution over circuits in $\mathcal{C}_n$. $\mathcal{O}$ is a distributional VBB obfuscator for $\mathcal{D}$ over $\mathcal{C}$ if:*

1. Functionality*: For each $n, C \in \mathcal{C}_n$ and $x \in \{0,1\}^n$,*

$$\Pr_{\mathcal{O},C'}[C'(x) = C(x)] \geq 1 - \mathtt{ngl}(n).$$

2. Slowdown*: For each $n, C \in \mathcal{C}_n$, the resulting C' can be evaluated in time $\mathtt{poly}(|C|, n)$.*

3. Security*: For each generic adversary $\mathcal{A}$ making at most m queries, there is a polynomial time simulator $\mathcal{S}$ such that $\forall n \in \mathbb{N}$, and each $D \in \mathcal{D}_n$ and each predicate P*

$$\left| \Pr_{\substack{C \leftarrow \mathcal{D}_n, \\ \mathcal{O}^{\mathcal{G}}, \mathcal{A}}} [\mathcal{A}^{\mathcal{G}}(\mathcal{O}^{\mathcal{G}}(C, 1^n)) = P(C)] - \Pr_{C \leftarrow \mathcal{D}_n, \mathcal{S}} [S^C(1^{|C|}, 1^n) = P(C)] \right| \leq \mathtt{ngl}(n).$$

Construction 2. *We now reiterate the construction from Bishop et al. adapted to use a random linear code for some prime $q = q(n)$.*

$\mathcal{O}$:

1. *Input $\mathbf{v} \in \{0, 1, \perp\}^n, q, g$ where g is a generator of the group $\mathbb{Z}_q^*$.*
2. *Sample $\mathbf{A} \in (\mathbb{Z}_q)^{2n \times n}, x_0 = 0, x_{1,\ldots,n-1} \leftarrow (\mathbb{Z}_q)^{n-1}$.*
3. *Sample $\mathbf{e} \in \mathbb{Z}_q^{2n}$ uniformly.*
4. *For $i = 0$ to $n - 1$:*
   (a) *If $v_i = 1$ set $e_{2i} = 0$.*
   (b) *If $v_i = 0$ set $e_{2i+1} = 0$.*
   (c) *If $v_i = \perp$ set $e_{2i} = 0, e_{2i+1} = 0$.*
5. *Compute $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{e}$.*
6. *Output $g^{\mathbf{y}}, \mathbf{A}$.*

Eval

1. *Input $g^{\mathbf{y}}, \mathbf{A}, x \in \{0,1\}^n$.*
2. *$\mathcal{I} = \{i \in [1...2n] \mid x_{\lfloor i/2 \rfloor} = (i \mod 2)\}$.*
3. *Compute $\mathbf{A}_{\mathcal{I}}^{-1}$. If none exists output $\perp$.*
4. *Output $g^{\mathbf{A}_{\mathcal{I},1}^{-1} \cdot \mathbf{y}} \stackrel{?}{=} g$.*

To state our security theorem we need to consider the transform from strings v over $\{0, 1, \perp\}$ to binary strings.

$$\mathsf{Bin}(v) = \mathbf{s} \text{ where } \begin{cases} s_i = 10 & \text{if } v_i = 1, \\ s_i = 01 & \text{if } v_i = 0, \\ s_i = 00 & \text{if } v_i = \perp . \end{cases}$$

Theorem 5. *Let $\ell \in \mathbb{Z}^+$ be a free parameter. Define $\mathcal{D}$ as the set of all distributions V such that $\mathsf{Bin}(V)$ is a distribution with $\alpha, (n-(\ell-1))$-entropy subsets. Then Construction 2 is VBB secure for generic $\mathcal{D}$ making at most m queries with distinguishing probability at most*

$$2\left(\frac{m(m+1)}{2}\left(\frac{1}{2^\alpha}+\frac{2}{q}\right)+\frac{\binom{2n}{n}}{q^\ell}\right).$$

Proof. Like the work of Bishop et al. [BKM+18, Theorem 16] the VBB security of the theorem follows by noting for any adversary $\mathcal{A}$ there exists a simulator S that initializes $\mathcal{A}$, provides them with $2n$ random handles (and simulates the interaction with $\mathcal{O}_r$) and outputs their output. By Theorem 1, the output of this simulator differs from the adversary in the real game by at most the above probability. □

6 Hardness of Decoding in the Standard Model

In this section, we consider whether decoding is hard for groups where the discrete logarithm problem is believed to be hard. We first examine hardness of decoding random linear codes in the exponent. In Appendix A we consider Reed-Solomon codes. Both results follow the same three part outline:

1. A theorem of Brands [Bra93] which says that if given a uniformly distributed $g^{\mathbf{y}}$ one can find $\mathbf{z}$ such that $g^{\langle \mathbf{y},\mathbf{z}\rangle}=1$ or equivalently that a vector $\mathbf{z}$ such that $\langle \mathbf{y},\mathbf{z}\rangle = 0$ then one can solve discrete log with the same probability. For a vector of length n and prime q, this problem is known as the $\mathtt{FIND}-\mathtt{REP}(n,q)$ problem.

2. A combinatorial lemma which shows conditions for a random $g^{\mathbf{y}}$ to be within some distance parameter c of a codeword with noticeable probability. That is, $\exists \mathbf{z} \in \mathbb{C}$ such that $\mathsf{dis}(g^{\mathbf{x}}, g^{\mathbf{z}}) \leq c$ (for the codeword space $\mathbb{C}$).

3. Let $\mathcal{O}$ be an oracle for bounded distance decoding. That is, given $g^{\mathbf{y}}$, $\mathcal{O}$ returns some $g^{\mathbf{z}}$ where $\mathsf{dis}(g^{\mathbf{z}}, g^{\mathbf{y}}) \leq c$ and $\mathbf{z} \in \mathbb{C}$. Recall that linear codes have known null spaces. Thus, if two vectors $g^{\mathbf{z}}$ and $g^{\mathbf{y}}$ match in more positions than the dimension of the code it is possible to compute a vector $\vec{\lambda}$ that is only nonzero in positions where $g^{\mathbf{z}_i} = g^{\mathbf{y}_i}$ and $\langle \vec{\lambda}, \mathbf{x}\rangle = \langle \vec{\lambda}, \mathbf{y}\rangle = 0$. If $\mathcal{O}$ works on a random point $g^{\mathbf{y}}$ it is possible to compute a vector $\vec{\lambda}$ in the null space of $\mathbf{y}$. This serves as an algorithm to solve the $\mathtt{FIND}-\mathtt{REP}$ and completes the connection to hardness of discrete log.

In this section we focus on a combinatorial lemma to establish point 2. In Appendix A, we present a similar result for Reed-Solomon codes improving prior work of Peikert [Pei06].

An (n,k,q)- random linear code, denoted $\mathbb{RL}(n,k,q)$, is generated by a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times k}$ that is independent and uniform elements of $\mathbb{Z}_q$. The code is the set of $\mathbf{Ax}$ for all vectors $\mathbf{x} \in \mathbb{F}_q^k$. We will consider noise vectors $\mathbf{e} \in \mathbb{F}_q$ where the Hamming weight of $\mathbf{e}$ denoted $\mathsf{wt}(\mathbf{e}) = t$ and the nonzero entries of $\mathbf{e}$ are uniformly distributed. That is, we consider $\mathbf{z} = \mathbf{Ax} + \mathbf{e}$.

Usually in coding theory the goal is *unique decoding*. That is, given some $\mathbf{y}$, if there exists some $\mathbf{z} \in \mathbb{C}$ such that $\mathsf{dis}(\mathbf{y},\mathbf{z}) \leq t$, the algorithm is guaranteed to return $\mathbf{y}$ and $\mathbf{z}$ is uniquely defined.

Our results consider algorithms that perform bounded distance decoding. Bounded distance decoding is a relaxation of unique decoding. For a distance t and a point $\mathbf{y} \in \mathbb{Z}_q^n$ a bounded distance decoding algorithm returns some $\mathbf{z} \in \mathbb{C}$ such that $\mathsf{dis}(\mathbf{y},\mathbf{z}) \leq t$. There is no guarantee that $\mathbf{z}$ is unique or is the point in the code closest to $\mathbf{y}$.

Problem $\texttt{BDDE} - \texttt{RL}(n, k, q, c, g)$, or Bounded Distance Decoding in the exponent of Random Linear Codes codes.

Instance Known generator g of $\mathbb{Z}_q^*$. Define $\mathbf{e}$ as a random vector of weight c in $\mathbb{Z}_q$. Define $g^{\mathbf{y}} = g^{\mathbf{Ax}+\mathbf{e}}$ where $\mathbf{A}, \mathbf{x}$ are uniformly distributed. Input is $g^{\mathbf{y}}, \mathbf{A}$.

Output Any codeword $g^{\mathbf{z}}$ where $\exists \mathbf{x} \in \mathbb{Z}_q^k$ such that $\mathbf{z} = \mathbf{Ax}$ and $\mathsf{dis}(\mathbf{x}, \mathbf{z}) \leq c$.

For a code $\mathbf{C}$ we define the distance between a point $\mathbf{y}$ and the code as the minimum distance between $\mathbf{y}$ and any codeword $\mathbf{c}$ in $\mathbf{C}$. Formally,

$$\mathsf{dis}(\mathbf{y}, \mathbf{C}) = \min_{\mathbf{c} \in \mathbf{C}} \mathsf{dis}(\mathbf{y}, \mathbf{c}).$$

Our proofs use the notion of *thickness* of a point with respect to a codespace and a radius. Consider some point $\mathbf{y}$ in the codespace and a radius r. The *thickness* of a point is the number of Hamming balls (of radius r) inflated around all codewords that cover $\mathbf{y}$. Specifically, define the set of points contained in a Hamming ball of radius r as $\Phi(r, \mathbf{z})$ for each codeword $\mathbf{z}$ in the code $\mathbf{C}$. Then define random variables $\varphi(r, \mathbf{z}, \mathbf{y})$ for each $\Phi(r, \mathbf{z})$ where $\varphi(r, \mathbf{z}, \mathbf{y}) = 1$ if $\mathbf{y} \in \Phi(r, i)$ and 0 otherwise. Then the thickness of $\mathbf{y}$ is

$$\mathsf{Thick}(r, \mathbf{C}, \mathbf{y}) = \sum_{\mathbf{z} \in \mathbf{C}} \varphi(r, \mathbf{z}, \mathbf{y}).$$

We now present the theorem of this section and our key technical lemma (Lemma 4), then prove the lemma and finally the theorem.

Theorem 6. *For positive integers n, k, c and q where $k < n \leq q$ and let g be a generator of $\mathbb{Z}_q^*$. If an efficient algorithm exists to solve $\texttt{BDDE} - \texttt{RL}(n, k, q, n-k-c, g)$ with probability ϵ, then an efficient randomized algorithm exists to solve the discrete log problem in the same group with probability at least*

$$\epsilon' = \epsilon \left(1 - \left(\frac{q^{n-k}}{\mathsf{Vol}(n, n-k-c, q)} + \frac{k}{q^{n-k}}\right)\right).$$

In particular, using a volume bound $\mathsf{Vol}(n, r, q) \geq \binom{n}{k} q^r (1 - n/q)$, we get

$$\epsilon' = \epsilon \left(1 - \left(\frac{q^c}{\binom{n}{k+c}(1 - \frac{n}{q})} + \frac{k}{q^{n-k}}\right)\right).$$

Lemma 4. *Let a Code $\mathbb{RL}_{\mathbf{A}}(n, k, q)$ be defined by matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times k}$, then*

$$\Pr_{\mathbf{y} \in \mathbb{F}_q^n, \mathbf{A}}[\mathsf{dis}(\mathbf{y}, \mathbb{RL}_{\mathbf{A}}(n, k, q)) > n - k - c] \leq \frac{q^{n-k}}{\mathsf{Vol}(n, n-k-c, q)} + \frac{1}{q^{n-k}}.$$

Proof of Lemma 4. A Random Linear Code $\mathbb{RL}_{\mathbf{A}}(n, k, q)$ has q^k codewords in a q^n sized codespace as long as $\mathbf{A}$ is full rank. The probability of $\mathbf{A}$ being full rank is at least $1 - k/q^{n-k}$ [FMR13, Lemma A.3]. The expected thickness of a code or $\mathbb{E}_{\mathbf{y}} \mathsf{Thick}(r, \mathbf{A}, \mathbf{y})$ is the average thickness over all points in the space. Expected thickness is the ratio of the sum of the volume of the balls and the size of the space itself. Note that this value can be greater than 1. A Hamming ball in this space can only be defined up to radius n. We give denote the expected thickness of the code as follows:

$$\mathbb{E}_{\mathbf{y}}(\mathsf{Thick}(r, \mathbf{A}, \mathbf{y})) = \frac{\mathsf{Vol}(n, r, q) \cdot q^k}{q^n} = \mathsf{Vol}(n, r, q) \cdot q^{k-n}$$

For $r = n - k - c$:

$$\mathbb{E}_{\mathbf{y}}(\mathsf{Thick}(n-k-c, \mathbf{A}, \mathbf{y})) \geq \mathsf{Vol}(n, n-k-c, q) \cdot q^{k-n}$$

For a point to be have Hamming distance from our code greater than $n-k-c$, its thickness must be 0. For the thickness of a point to be 0, it must deviate from the expected thickness by the expected thickness. We use this fact to bound the probability that a point is distance at least $n-k-c$. We require that each codeword is pairwise independent (that is, $\Pr_{\mathbf{A}}[c \in \mathbf{A} | c' \in \mathbf{A}] = \Pr_{\mathbf{A}}[c \in \mathbf{A}]$). In random linear codes, only generating matrices with dimension 1 are not pairwise independent. We have already restricted our discussion to full rank $\mathbf{A}$. Define an indicator random variable that is 1 when a point c is in the code. The pairwise independence of the code implies pairwise independence of these indicator random variables. With pairwise independent codewords, we use Chebyshev's Inequality to bound the probability of a random point being remote from a random code. We upper bound the variance of Thick by its expectation (since the random variable is nonnegative). In the below equations we only consider $\mathbf{A}$ where $\mathsf{Rank}(\mathbf{A}) = k$ but do not write this to simplify notation. Let $t = n - k - c$, then

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\mathbf{A}} \Pr_{\mathbf{y}}[\mathsf{dis}(\mathbf{y}, \mathbb{RL}_{\mathbf{A}}(n,k,q)) > t] \\
&\quad = \mathop{\mathbb{E}}_{\mathbf{A}} \Pr_{\mathbf{y}}[\mathsf{Thick}(t, \mathbf{A}, \mathbf{y}) = 0] \\
&\quad \leq \mathop{\mathbb{E}}_{\mathbf{A}} \left(\Pr_{\mathbf{y}} \left[|\mathsf{Thick}(t, \mathbf{A}, \mathbf{y}) - \mathbb{E}(\mathsf{Thick}(t, \mathbf{A}, \mathbf{y}))| > \mathbb{E}(\mathsf{Thick}(t, \mathbf{A}, \mathbf{y})) \right] \right) \\
&\quad \leq \mathop{\mathbb{E}}_{\mathbf{A}} \left(\frac{\mathrm{Var}_{\mathbf{y}}(\mathsf{Thick}(t, \mathbf{A}, \mathbf{y}))}{\mathbb{E}_{\mathbf{y}}(\mathsf{Thick}(t, \mathbf{A}, \mathbf{y}))^2} \right) \leq \mathop{\mathbb{E}}_{\mathbf{A}} \left(\frac{1}{\mathbb{E}_{\mathbf{y}}(\mathsf{Thick}(t, \mathbf{A}, \mathbf{y}))} \right) = \frac{q^{n-k}}{\mathsf{Vol}(n, n-k-c, q)}.
\end{aligned}$$

□

Proof of Theorem 6. Suppose an algorithm $\mathcal{F}$ solves $\mathtt{BDDE} - \mathtt{RL}(n, k, q, n-k-c, g)$ with probability ϵ. We show that $\mathcal{F}$ can be used to construct an $\mathcal{O}$ that solves $\mathtt{FIND} - \mathtt{REP}$.

$\mathcal{O}$ works as follows:

1. Input $\mathbf{y} = (y_1, \ldots, y_n)$ (where $\mathbf{y}$ is uniform over $\mathbb{Z}_q^n$).
2. Generate $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times k}$.
3. Run $\mathbf{z} \leftarrow \mathcal{F}(\mathbf{y}, \mathbf{A})$.
4. If $\mathsf{dis}(\mathbf{y}, \mathbf{z}) > n - k - c$ output $\perp$.
5. Let $\mathcal{I} = \{i | \mathbf{y}_i = \mathbf{z}_i\}$.
6. Construct parity check matrix of $\mathbf{A}_{\mathcal{I}}$, denoted $H_{\mathcal{I}}$.
7. Find some nonzero row of H_I, denoted $\mathbf{B} = (b_1, \ldots, b_{k+c})$ with associated indices I.
8. Output $\vec{\lambda}$ where $\vec{\lambda}_i = \mathbf{B}_{i'}$ for $i \in \mathcal{I}$ where i' represents the location of i in a sorted list with the same elements as $\mathcal{I}$ and 0 otherwise.

By Lemma 4, $(\mathbf{y}, \mathbf{A})$ is a uniform instance of $\mathtt{BDDE} - \mathtt{RL}(n, k, q, n-k-c, g)$ with probability at least $1 - (q^{n-k}/\mathsf{Vol}(n, n-k-c, q) + k * q^{-(n-k)})$. This means that $\mathcal{I} \geq k + c$. Note for $\mathbf{z}$ to be a codeword it must be that there exists some $\mathbf{x}$ such that $\mathbf{z} = \mathbf{A}\mathbf{x}$ and thus, the parity check matrix restricted to $\mathcal{I}$ is defined and there is some nonzero row. □

References

[ACEK17] Daniel Apon, Chongwon Cho, Karim Eldefrawy, and Jonathan Katz. Efficient, reusable fuzzy extractors from LWE. In *International Conference on Cyber Security Cryptography and Machine Learning*, pages 1–18. Springer, 2017.

[AGV09] Adi Akavia, Shafi Goldwasser, and Vinod Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In Omer Reingold, editor, *Theory of Cryptography*, volume 5444 of *Lecture Notes in Computer Science*, pages 474–495. Springer Berlin Heidelberg, 2009.

[BC10] Nir Bitansky and Ran Canetti. On strong simulation and composable point obfuscation. In *Advances in Cryptology–CRYPTO 2010*, pages 520–537. Springer, 2010.

[BKM+18] Allison Bishop, Lucas Kowalczyk, Tal Malkin, Valerio Pastro, Mariana Raykova, and Kevin Shi. A simple obfuscation scheme for pattern-matching with wildcards. In *Annual International Cryptology Conference*, pages 731–752. Springer, 2018.

[BLMZ18] James Bartusek, Tancrde Lepoint, Fermi Ma, and Mark Zhandry. New techniques for obfuscating conjunctions. Cryptology ePrint Archive, Report 2018/936, 2018. `https://eprint.iacr.org/2018/936`.

[BLP+13] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In *Proceedings of the 45th annual ACM symposium on Symposium on theory of computing*, pages 575–584. ACM, 2013.

[BMvT78] Elwyn Berlekamp, Robert McEliece, and Henk van Tilborg. On the inherent intractability of certain coding problems. *IEEE Transactions on Information Theory*, 24(3):384 – 386, May 1978.

[BPM+92] Peter F Brown, Vincent J Della Pietra, Robert L Mercer, Stephen A Della Pietra, and Jennifer C Lai. An estimate of an upper bound for the entropy of english. *Computational Linguistics*, 18(1):31–40, 1992.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

[Bra93] Stefan Brands. Untraceable off-line cash in wallet with observers. In *Annual International Cryptology Conference*, pages 302–318. Springer, 1993.

[BV14] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) lwe. *SIAM Journal on Computing*, 43(2):831–871, 2014.

[CD08] Ran Canetti and Ronny Ramzi Dakdouk. Obfuscating point functions with multibit output. In *Advances in Cryptology–EUROCRYPT 2008*, pages 489–508. Springer, 2008.

[CFP+16] Ran Canetti, Benjamin Fuller, Omer Paneth, Leonid Reyzin, and Adam Smith. Reusable fuzzy extractors for low-entropy distributions. In *Advances in Cryptology – EUROCRYPT*, pages 117–146. Springer, 2016.

[CG99] Ran Canetti and Shafi Goldwasser. An efficient threshold public key cryptosystem secure against adaptive chosen ciphertext attack. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 90–106. Springer, 1999.

[CKVW10] Ran Canetti, Yael Tauman Kalai, Mayank Varia, and Daniel Wichs. On symmetric encryption and point obfuscation. In *Theory of Cryptography, 7th Theory of Cryptography Conference, TCC 2010, Zurich, Switzerland, February 9-11, 2010. Proceedings*, pages 52–71, 2010.

[CW07] Qi Cheng and Daqing Wan. On the list and bounded distance decodability of reed–solomon codes. *SIAM Journal on Computing*, 37(1):195–209, 2007.

[Dau04] John Daugman. How iris recognition works. *Circuits and Systems for Video Technology, IEEE Transactions on*, 14(1):21 – 30, January 2004.

[Des92] Yvo Desmedt. Threshold cryptosystems. In *Advances in Cryptology – AUSCRYPT*, pages 1–14. Springer, 1992.

[DMQ13] Nico Döttling and Jörn Müller-Quade. Lossy codes and a new variant of the learning-with-errors problem. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT*, volume 7881 of *Lecture Notes in Computer Science*, pages 18–34. Springer, 2013.

[DORS08] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM Journal on Computing*, 38(1):97–139, 2008.

[ElG85] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory*, 31(4):469–472, 1985.

[Eli57] Peter Elias. List decoding for noisy channels. 1957.

[FKL18] Georg Fuchsbauer, Eike Kiltz, and Julian Loss. The algebraic group model and its applications. In *Advances in Cryptology – CRYPTO*, pages 33–62. Springer, 2018.

[FMR13] Benjamin Fuller, Xianrui Meng, and Leonid Reyzin. Computational fuzzy extractors. In *Advances in Cryptology-ASIACRYPT 2013*, pages 174–193. Springer, 2013.

[FRS16] Benjamin Fuller, Leonid Reyzin, and Adam Smith. When are fuzzy extractors possible? In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 277–306. Springer, 2016.

[GS98] Venkatesan Guruswami and Madhu Sudan. Improved decoding of reed-solomon and algebraic-geometric codes. In *Foundations of Computer Science, 1998. Proceedings. 39th Annual Symposium on*, pages 28–37. IEEE, 1998.

[Gur10] Venkatesan Guruswami. Introduction to coding theory - lecture 2: Gilbert-Varshamov bound. University Lecture, 2010.

[GVW15] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. *Journal of the ACM (JACM)*, 62(6):45, 2015.

[HRvD+16] Charles Herder, Ling Ren, Marten van Dijk, Meng-Day Yu, and Srinivas Devadas. Trapdoor computational fuzzy extractors and stateless cryptographically-secure physical unclonable functions. *IEEE Transactions on Dependable and Secure Computing*, 2016.

[IPS09] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Secure arithmetic computation with no honest majority. In *Theory of Cryptography Conference*, pages 294–314. Springer, 2009.

[MP13] Daniele Micciancio and Chris Peikert. Hardness of SIS and LWE with Small Parameters. In *Advances in Cryptology - CRYPTO 2013*, Lecture Notes in Computer Science. 2013.

[MZ11] Marcelo A Montemurro and Damián H Zanette. Universal entropy of word ordering across linguistic families. *PLoS One*, 6(5):e19875, 2011.

[NZ93] Noam Nisan and David Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, pages 43–52, 1993.

[Pei06] Chris Peikert. On error correction in the exponent. In *Theory of Cryptography Conference*, pages 167–183. Springer, 2006.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the thirty-seventh annual ACM Symposium on Theory of Computing*, pages 84–93, New York, NY, USA, 2005. ACM.

[Reg10] Oded Regev. The learning with errors problem (invited survey). In *Proceedings of the 2010 IEEE 25th Annual Conference on Computational Complexity*, pages 191–204. IEEE Computer Society, 2010.

[RS60] Irving S Reed and Gustave Solomon. Polynomial codes over certain finite fields. *Journal of the society for industrial and applied mathematics*, 8(2):300–304, 1960.

[Sha51] Claude E Shannon. Prediction and entropy of printed english. *Bell system technical journal*, 30(1):50–64, 1951.

[Sha79] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.

[Sho97] Victor Shoup. Lower bounds for discrete logarithms and related problems. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 256–266. Springer, 1997.

[SSF18] Sailesh Simhadri, James Steel, and Benjamin Fuller. Reusable authentication from the iris. 2018.

[WB86] Lloyd R Welch and Elwyn R Berlekamp. Error correction for algebraic block codes, December 30 1986. US Patent 4,633,470.

[WZ17] Daniel Wichs and Giorgos Zirdelis. Obfuscating compute-and-compare programs under lwe. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 600–611. IEEE, 2017.

A Decoding Reed Solomon Codes in the Exponent

The Reed-Solomon family of error correcting codes [RS60] have extensive applications in cryptography. For the field $\mathbb{F}_q$ of size q, a message length k, and code length n, such that $k \leq n \leq q$, define the Vandermonde matrix $\mathbf{V}$ where the ith row, $\mathbf{V}_i = [i^0, i^1, ..., i^k]$. The Reed Solomon Code $\mathbb{RS}(n, k, q)$ is the set of all points $\mathbf{Vx}$ where $\mathbf{x} \in \mathbb{F}_q^k$. Reed-Solomon Codes have known efficient algorithms for correcting errors. We note that for a particular vector $\mathbf{x}$ the generated vector $\mathbf{Vx}$ is a degree k polynomial with coefficients $\mathbf{x}$ evaluated at points $1, ..., n$.

The Berlekamp-Welch algorithm [WB86] corrects up to $(n - k + 1)/2$ errors in any codeword in the code. List decoding provides a weaker guarantee. The algorithm instead vectors a list containing codewords within a given distance to a point, the algorithm may return 0, 1 or many codewords [Eli57]. The list decoding algorithm of Guruswami and Sudan [GS98] can find all codewords within Hamming distance $n - \sqrt{nk}$ of a given word. Importantly, algorithms to correct errors in Reed-Solomon codes rely on nonlinear operations. Like with Random Linear Codes we consider hardness of constructing an oracle that performs bounded distance decoding.

Problem $\mathtt{BDDE} - \mathtt{RS}(n, k, q, c, g)$, or Bounded Distance Decoding in the exponent of Reed Solomon codes.

Instance A known generator g of $\mathbb{Z}_q^*$. Define $\mathbf{e}$ as a random vector of weight c in $\mathbb{Z}_q^*$. Define $g^{\mathbf{y}} = g^{\mathbf{Vx}+\mathbf{e}}$ where $\mathbf{x}$ is uniformly distributed. Input is $g^{\mathbf{y}}$.

Output Any codeword $g^{\mathbf{z}}$ where $\mathbf{z} \in \mathbb{RS}(n, k, q)$ such that $\mathsf{dis}(g^{\mathbf{y}}, g^{\mathbf{z}}) \leq c$.

Theorem 7. *For any positive integers n, k, c, and q such that $q \geq n^2/4$, $c \leq n+k$, $k \leq n$ and a generator g of the group $\mathcal{G}$, if an efficient algorithm exists to solve $\mathtt{BDDE} - \mathtt{RS}(n, q, k, n-k-c, g)$ with probability ϵ (over a uniform instance and the randomness of the algorithm), then an efficient randomized algorithm exists to solve the discrete log problem in $\mathcal{G}$ with probability*

$$\epsilon' \geq \begin{cases} \epsilon\left(1 - \frac{2q^c}{\binom{n}{k+c}}\right) & \frac{n^2}{2} \leq q \\ \epsilon\left(1 - \frac{cq^c}{\binom{n}{k+c}}\right) & \frac{n^2}{4} \leq q < \frac{n^2}{2} \end{cases}.$$

Proof. Like Theorem 6 the core of our theorem is a bound on the probability that a random point is close to a Reed-Solomon code.

Lemma 5. *For any positive integer $c \leq n - k$, define $\alpha = \frac{4q}{n^2}$, and any Reed-Solomon Code $\mathbb{RS}(n, k, q)$,*

$$\Pr_{\mathbf{y}}[\mathsf{dis}(\mathbf{y}, \mathbb{RS}(n, k, q)) > n - k - c] \leq \frac{q^c}{\binom{n}{k+c}} \alpha^{-c} \sum_{c'=0}^{c} \alpha^{c'}$$

where the probability is taken over the uniform choice of $\mathbf{y}$ from $\mathcal{G}^n$.

Proof of Lemma 5. A vector $\mathbf{y}$ has distance at most $n-k-c$ from a Reed-Solomon code if there is some subset of indices of size $k+c$ whose distance from a polynomial is at most $k-1$. To codify this notion we define a predicate which we call *low degree*. A set S consisting of ordered pairs $\{\alpha_i, x_i\}_i$ is low degree if the points $\{(\alpha_i, \log_g x_i)\}_{i \in S}$ lie on a polynomial of degree at most $k-1$. Define $\mathcal{S} = \{S \subseteq [n] : |S| = k + c\}$. For every $S \in \mathcal{S}$, define Y_S to be the indicator random variable for if S satisfies the low degree condition taken over the random choice of $\mathbf{y}$. Let $Y = \sum_{S \in \mathcal{S}} Y_S$.

For all $S \in \mathcal{S}, \Pr[Y_S = 1] = q^{-c}$, because any k points of $\{(\alpha_i, \log_g x_i)\}_{i \in S}$ define a unique polynomial of degree at most k. The remaining c points independently lie on that polynomial with probability $1/q$. The size of $\mathcal{S}$ is $|\mathcal{S}| = \binom{n}{k+c}$. Then by linearity of expectation, $\mathbb{E}[Y] = \binom{n}{k+c}/q^c$. Now we use Chebyshev's inequality,

$$\begin{aligned}\Pr_{\mathbf{y}}[\mathsf{dis}(\mathbf{y}, \mathbb{RS}(n,k,q)) > n-k-c] &= \Pr[Y=0] \\ &\leq \Pr[|Y - \mathbb{E}[Y]| \geq \mathbb{E}[Y]] \\ &\leq \frac{\mathrm{Var}(Y)}{\mathbb{E}[Y]^2}.\end{aligned}$$

It remains to analyze $\mathrm{Var}(Y) = \mathbb{E}[Y^2] - \mathbb{E}[Y]^2$. To analyze this variance we split into cases where the intersection of Y_S and $Y_{S'}$ is small and large. Consider two sets S and S' and the corresponding indicator random variables Y_S and $Y_{S'}$. If $|S \cap S'| < k$ then $\mathbb{E}[Y_S | Y_{S'} = \mathbb{E}[Y_S]$ and $\mathbb{E}[Y_S Y_{S'}] = \mathbb{E}[Y_S]\,\mathbb{E}[Y_{S'}]$. This observation is crucial for security of Shamir's secret sharing [Sha79]. For pairs S, S' where $|S \cap S'| \geq k$, we introduce a variable c' between 0 and c to denote $c' = |S \cap S'| - k$. For such S, S' instead of computing $\mathbb{E}[Y^2] - \mathbb{E}[Y]^2$ we just compute $\mathbb{E}[Y^2]$ and use this as a bound. For each c' we calculate $\mathbb{E}[Y_S Y_{S'}]$ where $|S \cap S| = k + c'$. The number of pairs can be counted as follows: $\binom{n}{k+c}$ choices for S, then $\binom{k+c}{c-c'}$ choices for the elements of S not in S' which determines the $k + c'$ elements that are in both S and S', and finally $\binom{n-k-c}{c-c'}$ to pick the remaining elements of S' that are not in S. So the total number of pairs is $\binom{n}{k+c}\binom{k+c}{c-c'}\binom{n-k-c}{c-c'}$. Using these observations, we can upper bound the variance $\mathrm{Var}(Y)$ for our random variable Y:

$$\begin{aligned}\mathrm{Var}(Y) &= \sum_{S,S' \in \mathcal{S}} (\mathbb{E}[Y_S Y_{S'}] - \mathbb{E}[Y_S]\,\mathbb{E}[Y_{S'}]) \\ &= \sum_{c'=0}^{c} \sum_{\substack{S,S' \in \mathcal{S} \\ |S \cap S'| = k+c'}} (\mathbb{E}[Y_S Y_{S'}] - \mathbb{E}[Y_S]\,\mathbb{E}[Y_{S'}]) \\ &\leq \sum_{c'=0}^{c} \sum_{\substack{S,S' \in \mathcal{S} \\ |S \cap S'| = k+c'}} (\mathbb{E}[Y_S Y_{S'}]) = \sum_{c'=0}^{c} \sum_{\substack{S,S' \in \mathcal{S} \\ |S \cap S'| = k+c'}} (\frac{1}{q^{2c-c'}})\end{aligned}$$

Here the last line follows by observing that for both Y_S and $Y_{S'}$ to be 1 they must both define the same polynomial. Since S and S' share $k + c'$ points, there are $(k + c) + (k + c) - (k + c') = k + 2c - c'$ points that must lie on the at most $k - 1$ degree polynomial, and any k points determine the polynomial, and the remaining $2c - c'$ points independently lie on the polynomial with probability $1/q$ then the probability

that this occurs is $1/q^{2c-c'}$. Continuing one has that,

$$\begin{aligned}
\mathrm{Var}(Y) &\leq \frac{1}{q^{2c}} \sum_{c'=0}^{c} \sum_{\substack{S,S' \in \mathcal{S} \\ |S \cap S'| = k+c'}} (q^{c'}) \\
&= \frac{1}{q^{2c}} \sum_{c'=0}^{c} (q^{c'} \binom{n}{k+c} \binom{k+c}{c-c'} \binom{n-k-c}{c-c'}) \\
&= \Big[\binom{n}{k+c} \frac{1}{q^c} \Big] \frac{1}{q^c} \sum_{c'=0}^{c} (q^{c'} \binom{n}{k+c} \binom{k+c}{c-c'} \binom{n-k-c}{c-c'}) \\
&= \frac{\mathbb{E}[Y]}{q^c} \sum_{c'=0}^{c} \left(q^{c'} \binom{k+c}{c-c'} \binom{n-k-c}{c-c'} \right)
\end{aligned}$$

We bound the size of $\binom{k+c}{c-c'}\binom{n-k-c}{c-c'}$ by observing that the sum of the top terms of the choose functions is n and the product of two values with a known sum is bounded by the product of their average, in this case $n/2$. We also use the upper bound of the choose function where $n^k \geq \binom{n}{k}$ to arrive at the bound that

$$q^c \sum_{c'=0}^{c} (q^{c'} \binom{k+c}{c-c'} \binom{n-k-c}{c-c'}) \leq \left(\frac{(n/2)^2}{q} \right)^c \sum_{c'=0}^{c} \Big(\frac{q}{(n/2)^2} \Big)^{c'}.$$

The proof then follows using our bound for variance by defining $\alpha = 4q/n^2$. This completes the proof of Lemma 5. □

The remainder of the proof is similar to the proof of Theorem 6. $\mathcal{A}$ works as follows: on input $\mathbf{y}$ where $\mathbf{y}$ is uniform over G^n immediately run $\mathcal{D}(g, \mathbf{y})$. By Lemma 5, $(g, \mathbf{v})$ is an instance of $\texttt{BDDE} - \texttt{RS}_{q,\mathcal{E},k,n-k-c}$ with probability at least

$$1 - \frac{q^c}{\binom{n}{k+c}} \alpha^{-c} \sum_{c'=0}^{c} \alpha^{c'}.$$

Then conditioned on this event, the instance is uniform, and $\mathcal{D}$ (with probability ϵ) outputs some $\mathbf{z}$ where $\mathsf{dis}(\mathbf{z}, \mathbf{y}) \leq n - k - c$. Take any $k+1$ indices $\mathcal{I} \subseteq [n]$ such that $\mathbf{y}_i = \mathbf{z}_i$ for $i \in E$. Then any k of the $\mathbf{y}_i$ interpolate to another one of the $\mathbf{y}_i$. We find the non-trivial Lagrange coefficients for the first k $\mathbf{y}_i$ call them λ_i such that $\prod_{i \in E} v_i^{\lambda_i} = 1$. Call the remaining point $\mathbf{y}_{k+1}$. let $\lambda_i = 0$ for $i \notin E$ and set λ_{k+1} to -1. Then $(\lambda_1, \ldots, \lambda_n)$ is a solution to $\texttt{FIND} - \texttt{REP}$.

The parameters in the Theorem follow when $1 \leq \alpha < 2$ by noting that

$$\alpha^{-c} \sum_{c'=0}^{c} \alpha^{c'} \leq \alpha^{-c}(c \cdot \alpha^c) = c.$$

Parameters in Theorem 7 follow in the case when $\alpha = 4q/n^2 \geq 2$ by noting that:

$$\alpha^{-c} \sum_{c'=0}^{c} \alpha^{c'} = \alpha^{-c} \Big(\frac{\alpha^{c+1} - 1}{\alpha - 1} \Big) = \Big(\frac{\alpha - \alpha^{-c}}{\alpha - 1} \Big) \leq 2.$$

This completes the proof of Theorem 7. □