Reduce and Prange: Revisiting Prange's Information Set Decoding for LPN and RSD

Jiseung Kim[1] and Changmin Lee[2]

[1] Jeonbuk National University, Jeonju, Rep. of Korea
jiseungkim@jbnu.ac.kr
[2] Korea Institute for Advanced Study, Seoul, Rep. of Korea
changminlee@kias.re.kr

Abstract. The learning parity with noise (LPN) problem has been widely utilized in classical cryptography to construct cryptographic primitives. Various variants of LPN have been proposed, including LPN over large fields and LPN with regular noise, depending on the underlying space and the noise regularity. These LPN variants have proven to be useful in constructing cryptographic primitives.

We propose an improvement to the Gaussian elimination attack, which is also known as Prange's information set decoding algorithm, for solving the LPN problem. Contrary to prevailing knowledge, we find that the Gaussian elimination attack is highly competitive and currently the best method for solving LPN over large fields. Our improvement involves applying partial Gaussian elimination repeatedly, rather than the whole Gaussian algorithm, which we have named the "Reduce and Prange's algorithm".

Moreover, we provide two applications of Reduce and Prange algorithms: One is the hybrid algorithm of ours and Berstein, Lange and Peters's algorithm at PQCrypto'08, and the other one is Reduce and Prange algorithm for LPN with regular noise.

Last, we provide a concrete estimation of the bit-security of LPN variants using our Reduce and Prange's frameworks. Our results show that the bit-security of LPN over $\mathbb{F}_q$ is reduced by 5-11 bits when $\log q = 128$ compared to previous analysis by Liu et al. (will appear at Eurocrypt'24). Furthermore, we show that our algorithm outperforms recent work by Briaud and Øygard (Eurocrypt'23) and Liu et al. for certain parameters. It reduces the bit-security of LPN with regular noise by 5-28 bits.

Keywords: LPN (over large fields), LPN with regular noise, Concrete security

1 Introduction

As the central problem of learning theory and coding theory, the learning parity with noise (LPN) problem has affected numerous cryptographic primitives such as secure arithmetic computations [4, 7, 14, 16, 17, 25, 33, 40, 44, 56, 62, 71], zero knowledge proofs [8, 29, 67] and more [1–3, 11, 22, 26, 32, 45, 46, 48, 50, 72, 73].

Problem 1 (Learning Parity with Noise (LPN)). Let m, n, t be positive integers and $\mathcal{R}$ be a ring. Let $\mathcal{C}$ be a probabilistic code generation algorithm such that $\mathcal{C}(m, n, \mathcal{R})$ returns a matrix $\mathbf{A} \in \mathcal{R}^{m \times n}$. Let $\chi(\mathcal{R}) = \{\chi_{m,t}\}_{m,t \in \mathbb{N}}(\mathcal{R})$ be a family of distributions over $\mathcal{R}^m$ that returns a vector in $\mathcal{R}^m$, where the number of nonzero coefficients in the vector is t.

The computational learning with parity problem with respect to parameters m, n and t involves obtaining a secret vector $\mathbf{s}$ given instances

$$(\mathbf{A}, \mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e} \bmod \mathcal{R})$$

where $\mathbf{A} \leftarrow \mathcal{C}(m, n, \mathcal{R}), \mathbf{s} \leftarrow \mathcal{R}^n$ and $\mathbf{e} \leftarrow \chi_{m,t}(\mathcal{R})$. We simply say the problem (m, n, t)-LPN problem over $\mathcal{R}$.[1]

In particular, for pseudorandom correlation generator (PCG) framework by [14], three rings have been used: fields $\mathbb{F}_2, \mathbb{F}_q$ and ring $\mathbb{Z}_{2^\lambda}$ for some $\lambda > 1$. Recent work [49] claimed that (m, n, t)-LPN over $\mathbb{Z}_{2^\lambda}$ can be reduced by $(m, n, t/2)$-LPN over $\mathbb{F}_2$, which leads to suffice to analyze the security of LPN over both fields $\mathbb{F}_2$ and $\mathbb{F}_q$, respectively. If $\mathcal{R} = \mathbb{F}_2$, then it is called the standard LPN problem or simply LPN. If $\mathcal{R} = \mathbb{F}_q$ for some large prime power q, then we call it 'LPN over large fields', where it is a natural variant of the (standard) LPN.

The recent primitives also assume the regularity of the error distribution. Such a variant of LPN is called, LPN with regular noise (for short regular-LPN) defined as follows:

Problem 2 (LPN with regular noise, regular-LPN). Let m, n, t, β be positive integers with $m = t \cdot \beta$ and $\mathcal{R}$ be a ring. Let $\mathcal{C}$ be a probabilistic code generation algorithm such that $\mathcal{C}(m, n, \mathcal{R})$ returns a matrix $\mathbf{A} \in \mathcal{R}^{m \times n}$. Let $\tau(\mathcal{R}) = \{\tau_{m,t}\}_{m,t \in \mathbb{N}}(\mathcal{R})$ be a family of distributions over $\mathcal{R}^m$. Given m, t, a distribution $\tau_{m,t}$ returns a vector $\mathbf{e} = (\mathbf{e}_1, \ldots, \mathbf{e}_t) \in \mathcal{R}^m$ such that $\mathbf{e}_i \in \mathcal{R}^\beta$ is of Hamming weight $|\mathbf{e}_i| = 1$ for $1 \leq i \leq t$.

The computational LPN with regular noise involves obtaining a secret vector $\mathbf{s}$ given instances

$$(\mathbf{A}, \mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e} \bmod \mathcal{R})$$

where $\mathbf{A} \leftarrow \mathcal{C}(m, n, \mathcal{R}), \mathbf{s} \leftarrow \mathcal{R}^n$ and $\mathbf{e} \leftarrow \tau_{m,t}(\mathcal{R})$. We simply say the problem (m, n, t)-LPN problem over $\mathcal{R}$.

The use of PCG-like protocols would provide significant improvements in the design of efficient cryptographic primitives with practical applications [7,8,15,17, 19,25,27,28,30,31,42,47,57,60–62,65–71]. The PCG-like protocols additionally

[1] Following the previous work [16,17,49], we adapt the definition for our purpose. We note that the definition of LPN originally states that the goal is to find $\mathbf{s}$ using oracle access to $\mathcal{O}$, which returns a LPN instance. To be specific, we only consider an LPN where the number of oracle queries are limited to m. (The problem can be considered as a variant of decoding linear code.) This constraint is reasonable because most cryptographic primitives built on LPN use the parameter regime where the number of oracle queries are bounded.

employ a low-noise setting that $t/m = 1/n^{\varepsilon}$ for some constant $\varepsilon > 0$ with bounded samples.

However, most *concretely efficient* cryptographic applications built from the low-noise LPN or its variants have only a few cryptanalysis despite its various applications. Although Blum-Kalai-Wasserman (BKW) algorithm [12] is currently the most effective attack, BKW works for unbounded samples. Therefore, it is not suitable for all scenarios. Consequently, accurately estimating the concrete security level of the cryptographic primitives employed is crucial for their secure use. The concrete LPN parameter regime used in most protocols and their real world applications is originated by Boyle et al. [14], but the following work by Liu et al. [49] claimed that the analysis of [14] is oversimplified, which implies the time cost of actual attacks are underestimated. Indeed, the paper by Canteaut and Chabaud [20] argues that Gaussian elimination, a step commonly used in LPN cryptanaysis, is computationally expensive. Thus, Liu et al. initiate to provide a more accurate analysis for LPN over rings used in the pseudorandom correlation generator paradigm.

Analysis in [49]. We briefly summarize the results of the recent analysis [49] to easily introduce our contribution.

- For the parameter regime $\mathcal{R} = \mathbb{F}_q$ with prime power $q = n^{w(1)}$, $t = o(m)$ and $(1+\beta) \cdot n \leq m = poly(n)$ for some constant β, the statistical decoding including its 2.0 variant [23] needs more cost than Prange's algorithm [59].
- For the parameter setup, the information set decoding and its variants outperform other algorithms.
- For $\mathcal{R} = \mathbb{Z}_{2^\lambda}$ for some $\lambda > 1$, (m, n, t)-LPN over $\mathcal{R}$ can be solved through $(m, n, \frac{2^{\lambda-1}}{2^\lambda - 1} \cdot t)$-LPN over $\mathbb{F}_2$.

Analysis in [18, 37]. The work proposed by [18] describes a novel algebraic attack that exploits regular noise distributions on $\mathcal{R} = \mathbb{F}_q$ for any $q \geq 2$. Specifically, the regular noise distribution is used to generate multivariate polynomials, which can reduce the concrete security of LPN with regular noise. Subsequently, [37] modified the well-known algorithms for solving LPN into specified algorithms for solving LPN with regular noise on $\mathcal{R} = \mathbb{F}_2$. They mainly exploit the specific structure derived from the regular structure of the error vector. These results highlight the importance of addressing the regular noise distribution in LPN.

According to the [18,37,49,53], it is observed that when the field size is sufficiently large, Prange's algorithm has a comparable cost to other algorithms. [21] observes that when t is sublinear of n, then Prange's algorithm provides similar performance to improved ISD algorithms. In a nutshell, Prange's algorithm repeatedly conducts the Gaussian elimination algorithm until getting error-free samples.

Consequently, it implies that to get better LPN analysis over a large field, two approaches are necessary: 1) reduce the number of iterations and 2) reduce the cost of the Gaussian elimination algorithm.

1.1 Our Contribution

In this paper, we revisit Prange's ISD algorithm to improve its performance by reducing the cost of the Gaussian elimination part. We named the modified Prange's algorithm as Reduce and Prange algorithm (RP). The algorithm is quite effective for several LPN and its variants. As applications of RP algorithm, we further provide three algorithms:

- hybrid-RP: Hybrid algorithm that combines RP and the algorithm by Bernstein, Lange, and Peters [10].
- regular-RP: RP for specifying regular-LPN.

We implement a SageMath [64] script to search for near-optimal parameterization for our attacks. Although our approach may appear straightforward, our experimental results indicate that it can surpass existing methods for solving the LPN problem over large fields in PCG parameter settings. The estimated results for various parameters and rings are given by Table 1 and Table 2. Indeed, the bit-security of LPN over $\mathbb{F}_{2^{128}}$ is reduced by 5-11 bits when $\log q = 128$ compared to [49].

For regular-LPN, the results show that our algorithm outperforms the recent work proposed in [18, 37] for some parameter settings with relatively small n. Specifically, the bit-security is reduced by 5-28 bits for small (m, n) and $\log q = 128$. However, when t/n is sufficiently small, the approach presented in [18] outperforms regular-RP. For detailed numerical results on solving LPN with regular noise, please refer to Table 3.

1.2 Attack Idea

The RP algorithm starts from a simple observation: The concrete time complexity of the Gaussian elimination cannot be disregarded for a specific parameter regime although it asymptotically takes a polynomial time in n. It might seem counter-intuitive, but it underscores the importance of considering the practical aspects of the algorithm.

To be more specific, while the Gaussian elimination operation takes a time complexity of only $O(n^\omega)$ for the linear algebra constant ω, which is typically ignored in the context of asymptotic analysis, its concrete time complexity plays a significant role in estimating bit-security. For instance, we consider the $(1024, 652, 57)$-LPN over $\mathbb{F}_q$. According to [49], the bit-size of the total cost associated with this problem is reported to be 111, with the Gaussian elimination accounting for a bit-size of 23. This highlights that the Gaussian elimination contributes to approximately one-fifth of the total complexity involved in solving the problem.

The main idea for RP is to reduce the cost of the Gaussian elimination by combining the reduction strategy with Prange's algorithm. To be precise, the basic concept of RP is to convert (m, n, t)-LPN instances into (m_1, n_1, t_1)-LPN instances with $m \geq m_1, n \geq n_1$ and $t \geq t_1$ and solve the (m_1, n_1, t_1)-LPN problem. As a natural extension, this technique can be iteratively applied for

multiple rounds, further improving the overall efficiency of the algorithm. For the detailed algorithm, we refer Section 2.

Hybrid-RP: Combine ours with [10]. RP and [10] are motivated by heavy computational costs derived from the Gaussian elimination, so focusing on reducing the cost of the Gaussian elimination. The main idea of [10] is to reuse existing pivots to efficiently deploy the Gaussian elimination for (large) matrix.

On the other hand, RP algorithm begins by guessing positions of the zero value in the error vector. To illustrate, suppose that we guess N_1 positions, and the algorithm works if we can accurately guess all these N_1 positions. In case of incorrect guessing, we again attempt to guess the *whole* N_1 *positions*. By applying the reuse technique in [10], we then only guess $N_1' < N_1$ positions. This adaptation, leveraging the reusable property, can decrease the cost of our algorithm.

Consequently, by combining two algorithms, we propose a new algorithm such that 1) guessing N_1 positions, and 2) reusing some pivots, and 3) running the Gaussian elimination on a small matrix until the guess is correct. This hybrid algorithm is expected to outperform the individual methods. The detailed description of the hybrid algorithm is given by Section 2.4

Regular-RP: RP for regular-LPN. Using the regularity of the error vector $\mathbf{e}$, we modify the guessing probability in RP. Then, the other steps are identical to the original RP. The detailed algorithm is given by Section 3. It can also be optimized by combining [10] and regular-RP as in hybrid-RP, called the hybrid-regular-RP.

Notations. We represent vectors and matrices using boldface type. For any integer q, let $\mathbb{F}_q$ be a finite field of elements q. Let χ be a probability distribution defined on some finite set. Accordingly, the notation $s \leftarrow \chi$ denotes an element s sampled from the distribution χ. For a finite set S, $s \leftarrow S$ denotes an element s sample from the uniform distribution defined over S. Furthermore, we denote $\mathbf{0}^n$ as an n-dimensional zero vector for any positive integer n. The inner product between the two vectors $\mathbf{a}$ and $\mathbf{b}$ is denoted by $\langle \mathbf{a}, \mathbf{b} \rangle$. In certain cases, we use the shorthand notation $\mathbf{a} \cdot \mathbf{b}$ for simplicity.

2 Reduce and Prange Technique

This section introduces a new approach for solving LPN over large fields, simply called the Reduce and Prange. For the algorithm description, we suppose that LPN instances of the form $(\mathbf{A}, \mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e} \bmod q)$ with $\mathbf{A} \in \mathbb{F}_q^{m \times n}, \mathbf{s} \in \mathbb{F}_q^n$ and $\mathbf{e} \leftarrow \chi_{t,m}$ are given.

2.1 Prange's Information Set Decoding

We first provide a brief overview of Prange's information set decoding for solving the LPN problem. The algorithm can be informally divided into two main parts:

collecting linear polynomials with a common root and solving the resulting linear system to recover the secret vector $\mathbf{s}$.

More precisely, we denote the n-dimensional vector as $\mathbf{x}$. For each LPN instance $(\mathbf{a}_i, b_i)$, where $b_i = \langle \mathbf{a}_i, \mathbf{s} \rangle + e_i \bmod \mathcal{R}$ and $\mathbf{a}_i$ is an n-dimensional vector, we define f_i as an n-variate linear polynomial of the form

$$f_i(\mathbf{x}) = b_i - \langle \mathbf{a}_i, \mathbf{x} \rangle \bmod \mathcal{R}.$$

If $e_i = 0$, then $\mathbf{s}$ is a root of f_i. We call such polynomials *error-free* polynomials. In summary, Prange's algorithm collects n error-free polynomials and recovers the secret $\mathbf{s}$ with high probability through the Gaussian elimination.

It is important to note that the probability of collecting n error-free polynomials is $\frac{\binom{m-n}{t}}{\binom{m}{t}}$. Therefore, the algorithm ensures that the desired secret vector can be recovered with $\frac{\binom{m}{t}}{\binom{m-n}{t}}$ iterations.

Therefore, the time complexity of Prange's ISD algorithm is estimated by

$$O\left(\frac{\binom{m}{t}}{\binom{m-n}{t}} \cdot n^{\omega}\right)$$

where n^{ω} is a cost of the Gaussian elimination with the linear algebra constant $2 \leq \omega \leq 3$.

2.2 Reduce and Prange's ISD

In terms of concrete security estimation, the time complexity of the Gaussian Elimination, $O(n^{\omega})$, has a big portion in Prange's ISD algorithm. For example, finite regime estimate of security level in [49] for solving (m, n, t)-LPN over $\mathbb{F}_{2^{128}}$ with $(m, n, t) = (1024, 652, 106)$ is estimated by 194. To be more precise, the logarithmic scale of the number of iterations required to gather n error-free polynomials is approximately 171, while that of performing Gaussian elimination is approximately 23. Hence, the cost of Gaussian elimination should not be ignored in concrete estimations. If one can reduce the Gaussian elimination costs, then it implicitly yields to decrease in the total cost of the algorithm.

The intuition of the attack is that we first guess N_1 error-free samples to obtain LPN samples themselves of less dimension. We then apply Prange's algorithm to the LPN of $n - N_1$ dimensions.

Suppose that LPN instances of the form $(\mathbf{A}, \mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e} \bmod q)$ with $\mathbf{A} \in \mathbb{F}_q^{m \times n}, \mathbf{s} \in \mathbb{F}_q^n$ and $\mathbf{e} \leftarrow \chi_{t,m}$. We let $\mathbf{a}_i$ (Resp. b_i and e_i) denote the i-th vector of $\mathbf{A}$ (Resp. $\mathbf{b}$ and $\mathbf{e}$).

Step 1. Guess the $N_1 (< n)$ zero positions in $\mathbf{e}$. For easy explanation, we will assume that the last N_1 positions in $\mathbf{e}$ are zero. Let $\mathbf{A}_{top}$ and $\mathbf{A}_{bot}$ be a submatrix of $\mathbf{A}$ which consists of the first $m - N_1$ rows of $\mathbf{A}$ and the last N_1 rows, respectively. Analogously to the notation, we define $\mathbf{b}_{top}$, $\mathbf{b}_{bot}$, $\mathbf{e}_{top}$, and $\mathbf{e}_{bot}$.

The above assumption then says that $\mathbf{e}_{bot} = \mathbf{0}$. Under the notation, it holds that

$$\begin{pmatrix}\mathbf{b}_{top}\\ \mathbf{b}_{bot}\end{pmatrix} = \begin{pmatrix}\mathbf{A}_{top}\\ \mathbf{A}_{bot}\end{pmatrix}\cdot\mathbf{s} + \begin{pmatrix}\mathbf{e}_{top}\\ \mathbf{0}\end{pmatrix} \bmod q$$

We then apply the Gaussian elimination algorithm to obtain LPN of less dimension. For this purpose, We parse $\mathbf{s}$ to $(\mathbf{s}^1\|\mathbf{s}^2)$ such that $\mathbf{s}^1 \in \mathbb{F}_q^{n-N_1}$ and $\mathbf{s}^2 \in \mathbb{F}_q^{N_1}$. Similarly to the $\mathbf{s}$, we split $\mathbf{A}_{top}$ (Resp. $\mathbf{A}_{bot}$) into $(\mathbf{A}_{top}^{left}\|\mathbf{A}_{top}^{right})$ (Resp. $(\mathbf{A}_{bot}^{left}\|\mathbf{A}_{bot}^{right})$) as well. Assuming that the matrix $\mathbf{A}_{bot}^{right}$ is invertible over $\mathbb{F}_q$, (If not, we change the order of columns until $\mathbf{A}_{bot}^{right}$ is invertible), one can get the following LPN samples:

$$\begin{aligned}
&\left(\mathbf{I}_{m-N_1}\| -\mathbf{A}_{top}^{right}\cdot(\mathbf{A}_{bot}^{right})^{-1}\right)\cdot\begin{pmatrix}\mathbf{b}_{top}\\ \mathbf{b}_{bot}\end{pmatrix}\\
&= \left(\mathbf{I}_{m-N_1}\| -\mathbf{A}_{top}^{right}\cdot(\mathbf{A}_{bot}^{right})^{-1}\right)\cdot\left(\begin{pmatrix}\mathbf{A}_{top}\\ \mathbf{A}_{bot}\end{pmatrix}\cdot\mathbf{s} + \begin{pmatrix}\mathbf{e}_{top}\\ \mathbf{0}\end{pmatrix}\right) \bmod q\\
&= \underbrace{(\mathbf{A}_{top}^{left} - \mathbf{A}_{top}^{right}\cdot(\mathbf{A}_{bot}^{right})^{-1}\cdot\mathbf{A}_{bot}^{left})}_{=:\mathbf{A}'}\cdot\mathbf{s}^1 + \mathbf{e}_{top} \bmod q.
\end{aligned}$$

Complexity. In each guessing, this algorithm computes the matrix $\mathbf{A}'$, which consists of one matrix inversion and two matrix multiplications so it takes in time $\mathcal{C}_{m,N_1} := O(N_1^\omega + N_1^2\cdot(m-N_1) + N_1\cdot(m-N_1)\cdot(n-N_1))$, where ω is the linear algebra constant. The probability that attackers can correctly guess N_1 positions is $\mathcal{P}_{m,t,N_1} := \frac{\binom{m-t}{N_1}}{\binom{m}{N_1}}$, which directly implies that the time complexity to get LPN samples of dimension $n-N_1$ is $\frac{1}{\mathcal{P}_{m,t,N_1}}\cdot\mathcal{C}_{m,N_1}$.

Step 2. For ease of representation, we assume that LPN instances $(\mathbf{A},\mathbf{b})$ such that $\mathbf{b} = \mathbf{A}\cdot\mathbf{s}+\mathbf{e} \bmod q$ are given, where $\mathbf{b}\in\mathbb{F}_q^{m-N_1}$, $\mathbf{A}\in\mathbb{F}_q^{(m-N_1)\times(n-N_1)}$, $\mathbf{s}\in\mathbb{F}_q^{n-N_1}$ and $\mathbf{e}\in\chi_{t,m-N_1}$.

The second step just runs Prange's algorithm. That is, one first collects $n-N_1$ error-free polynomials from given $m-N_1$ LPN instances. Then, one can recover the secret $\mathbf{s}$ via the Gaussian elimination algorithm.

Complexity. It is obvious to compute the complexity of Prange's algorithm for solving LPN over with dimension $n-N_1$, samples $m-N_1$ and t nonzero:

$$\mathcal{S}_{m-N_1,n-N_1,t} \overset{def}{=} \frac{\binom{m-N_1}{n-N_1}}{\binom{m-N_1-t}{n-N_1}}\cdot(n-N_1)^\omega = \frac{1}{\mathcal{P}_{m-N_1,t,n-N_1}}\cdot(n-N_1)^\omega,$$

where the linear algebra constant ω. Putting it together, the LPN problem is solved within

$$\frac{1}{\mathcal{P}_{m,t,N_1}}\cdot(\mathcal{C}_{m,N_1} + \mathcal{S}_{m-N_1,n-N_1,t})$$

In the complexity, the term $\frac{1}{\mathcal{P}_{m,t,N_1}} \cdot \mathcal{S}_{m-N_1,n-N_1,t} = \frac{\binom{m}{n}}{\binom{m-t}{n}} \cdot (n-N_1)^\omega$ is a dominating term. Compared to the original Prange algorithm $\frac{\binom{m-t}{n}}{\binom{m}{n}} \cdot n^\omega$, the fractional part is shared but one can see that the multiplier factor, $(n-N_1)^\omega$, is smaller for the Reduce and Prange algorithm. This can be expected to lower the overall complexity of the algorithm.

2.3 Iterative Reduce and Prange Technique

'Step 1' in Section 2 can be interpreted as a self-reduction of LPN. As a natural extension, one can run Step 1 reduction repeatedly to get LPN samples of less dimension. Thereafter, one can solve the reduced LPN problems by Step 2. With respect to the number of reductions, we call this algorithm the level-k Reduce and Prange algorithm.

For instance, suppose $k = 2$, the level-2 Reduce and Prange algorithm corresponds as follows: First one can get LPN samples of dimension $n - N_1 - N_2$ from the $n - N_1$ dimensional LPN samples, and then it can be solved via the Prange algorithm on the less dimension. The interesting fact is that since this auxiliary step proceeds after the first step occurs, the overall time complexity for level-2 Reduce and Prange algorithm is given as

$$\frac{1}{\mathcal{P}_{m,t,N_1}} \cdot \left(\mathcal{C}_{m,N_1} + \frac{1}{\mathcal{P}_{m-N_1,t,N_2}} \left(\mathcal{C}_{m-N_1,N_2} + \mathcal{S}_{m-N_1-N_2,n-N_1-N_2,t} \right) \right).$$

As a generalization, the time complexity for level-k Reduce and Prange algorithm is then obviously computed. For ease of exposition, we define notations $\mathcal{P}(j), \mathcal{C}(j), \mathcal{S}$ as follows:

$$\begin{aligned} \mathcal{P}(j) &= \mathcal{P}_{(m-\sum_{i=1}^{j-1} N_i),t,N_j} \\ \mathcal{C}(j) &= \mathcal{C}_{(m-\sum_{i=1}^{j-1} N_i),N_j} \\ \mathcal{S} &= \frac{1}{\mathcal{P}_{(m-\sum_{i=1}^{k} N_i),t,(n-\sum_{i=1}^{k} N_i)}} \cdot \left(n - \sum_{i=1}^{k} N_i \right)^\omega \end{aligned}$$

Then, under the notation, the total cost of our algorithm is

$$\mathcal{T} = \frac{1}{\mathcal{P}(1)} \left(\mathcal{C}(1) + \frac{1}{\mathcal{P}(2)} \left(\mathcal{C}(2) + \cdots + \frac{1}{\mathcal{P}(k)} \left(\mathcal{C}(k) + \mathcal{S} \right) \right) \right)$$

To simplify it, we also define $\mathcal{P}'(j)$ as $\mathcal{P}'(j) = \prod_{i=1}^{j} \mathcal{P}(i)$. We then denote the total cost as

$$\mathcal{T} = \sum_{j=1}^{k} \frac{\mathcal{C}(j)}{\mathcal{P}'(j)} + \frac{\mathcal{S}}{\mathcal{P}'(k)}.$$

To (approximately) optimize the cost, we investigate the $\mathcal{T}$ in two parts. On the one hand, we point out that the $\mathcal{P}'(j)$ equals to $\frac{\binom{m-t}{\sum_{i=1}^{j} N_i}}{\binom{m}{\sum_{i=1}^{j} N_i}}$. It means that the first term $\frac{\mathcal{C}(1)}{\mathcal{P}'(1)}$ is only determined by the N_1. Inductively, if the $\sum_{i=1}^{j-1} N_i$ is fixed once, the term $\frac{\mathcal{C}(j)}{\mathcal{P}'(j)}$ is only determined by the N_j.

On the other hand, the last term $\frac{\mathcal{S}}{\mathcal{P}'(k)}$ is identical to $(n - \sum_{i=1}^{j} N_i)^{\omega} \cdot \frac{\binom{m}{n}}{\binom{m-t}{n}}$, of which the fraction is an invariant factor with respect to k. This means that the cost of the last term is written as $\frac{\binom{m-t}{n}}{\binom{m}{n}} \cdot c$ for some c. The term c gets smaller as $\sum_{i=1}^{j} N_i$ gets larger.

Since the overall cost of a positive addition will be optimized when all values are similar, we let the threshold be $\frac{\binom{m-t}{n}}{\binom{m}{n}} \cdot \Delta$ for some integer $\Delta \in \mathbb{Z}$ and choose the largest integer N_i such that each $\frac{\mathcal{C}(i)}{\mathcal{P}'(i)}$ term is less than the threshold, starting at $i = 1$. If there is no integer N_j for an index j, we let $N_j = n - \sum_{i=1}^{j-1} N_i$ and operate the Prange's algorithm of LPN of dimension N_j. Then the total cost is given as $\sum_{i=1}^{j-1} \frac{\mathcal{C}(j)}{\mathcal{P}'(j)} + \frac{\binom{m}{n}}{\binom{m-t}{n}} N_j^{\omega}$.

We try it for all Δ smaller than $n^{2.8}$, and we give the most optimized cost. The full algorithm for estimating the Reduce and Prange will be given by Algorithm 1.

Algorithm 1 Reduce and Prange Estimation

Input: LPN parameters (m, n, t)
Output: The total cost of the Reduce and Prange algorithm.
Set $Threshold = \frac{\binom{m}{n}}{\binom{m-t}{n}}$
Set $Cost = Threshold \cdot n^{2.8}$
for $\Delta = 1$ up to $n^{2.8}$ **do**
  Set $idx = 1$
  while $\exists N_{idx}$ such that $\frac{\mathcal{C}(idx)}{\mathcal{P}'(idx)} < \Delta \cdot Threshold$ **do**
    Set largest N_{idx} such that $\frac{\mathcal{C}(idx)}{\mathcal{P}'(idx)} < \Delta \cdot Threshold$
    Set $idx = idx + 1$
  end while
  Set $N_{idx} = n - \sum_{i=1}^{idx-1} N_i$
  $Cost = \min\left\{Cost, \sum_{i=1}^{idx-1} \frac{\mathcal{C}(i)}{\mathcal{P}'(i)} + Threshold \cdot N_{idx}^{2.8}\right\}$
end for
Return $Cost$ and $\{N_i\}_{i=1}^{idx}$

2.4 Hybrid-RP: Combine [BLP08] with Reduce and Prange

Bernstein, Lange and Peters [10] proposed a method for solving the McEliece cryptosystem, refining the Stern's attack algorithm [63]. Similar to our approach,

their algorithm includes a step that iteratively searches for certain non-zero coordinates. A key strategy they employed involves the technique of reusing existing pivots, which, as the name suggests, means re-applying some coordinates from the previous iteration.

On the other hand, RP first guesses N_1 zero positions in the vector $\mathbf{e}$. If one correctly guesses, then go to the next step. Otherwise, it attempts to re-guess "the whole N_1 zero positions" in the noise vector $\mathbf{e}$. Each iteration step requires a considerable computation cost, denoted as $\mathcal{C}_{m,N_1}$. We propose a hybrid algorithm, called a hybrid-RP, which incorporates the reusing technique to lessen this cost.

Specifically, we define $\mathbf{e}_{bot} \in \mathbb{Z}^{N_1}$ as the vector replaced in the first iteration. While $\mathbf{e}_{bot}$ is not entirely zero, it would contain several zero coordinates. We assume $\mathbf{e}_{bot}$ has N_1' zero coordinates with probability

$$\mathcal{P}'_{N_1,N_1',m,t} = \frac{\binom{m-t}{N_1'}\binom{t}{N_1-N_1'}}{\binom{m}{N_1}}.$$

We divide $\mathbf{e}_{bot}$ into two parts: $\mathbf{e}_1 \in \mathbb{Z}^{N_1-N_1'}$ and $\mathbf{e}_2 \in \mathbb{Z}^{N_1'}$, where $\mathbf{e}_{bot} = (\mathbf{e}_1|\mathbf{e}_2)$ and $\mathbf{e}_2 = \mathbf{0}$.

Under the notations, after the first step, LPN instances can be expressed as:

$$\mathbf{b}' = \mathbf{A}' \cdot \mathbf{s}^1 + \mathbf{A}'' \cdot \mathbf{e}_1 + \mathbf{e}_{top},$$

where $\mathbf{A}'' \in \mathbb{Z}^{(m-N_1)\times(N_1-N_1')}$. As in [10], our strategy is also to guess only $N_1 - N - 1'$ zero positions in the updated noise vector to eliminate the $\mathbf{e}_1$ vector. Consequently, the complexity for this updated iteration step is $\mathcal{C}_{m-N_1,N_1-N_1'}$.

On the other hand, we note that since $\mathbf{e}_1$ is non-zero and $\mathbf{e}$ has t non-zero coordinates, $\mathbf{e}_{top}$ consists of $t' = t - (N_1 - N_1')$ non-zero entries. This means the step's success probability is $\mathcal{P}_{m-N_1,t',N_1-N_1'}$. Therefore, by incorporating the reuse of coordinates, the algorithm for reducing N_1 coordinates concludes with a cost of:

$$\frac{1}{\mathcal{P}'_{N_1,N_1',m,t}} \cdot \left(\mathcal{C}_{m,N_1} + \frac{\mathcal{C}_{m-N_1,N_1-N_1'}}{\mathcal{P}_{m-N_1,t',N_1-N_1'}}\right). \tag{1}$$

For simplicity, we denote

$$\mathcal{C}'_{m,N_1,t'} = \mathcal{C}_{m,N_1} \cdot \mathcal{P}_{m-N_1,t',N_1-N_1'} + \mathcal{C}_{m-N_1,N_1-N_1'}.$$

Moreover, it is also observed that

$$\mathcal{P}'_{N_1,N_1',m,t} \cdot \mathcal{P}_{m-N_1,t',N_1-N_1'} = \mathcal{P}_{m,t,N_1}.$$

Therefore, under the notation, Equation (1) can be written as $\frac{\mathcal{C}'_{m,N_1,t'}}{\mathcal{P}_{m,t,N_1}}$. Compared to the original computational cost of $\frac{\mathcal{C}_{m,N_1}}{\mathcal{P}_{m,t,N_1}}$, the factor of $1/\mathcal{P}_{m,t,N_1}$ shifts from $\mathcal{C}_{m,N_1}$ to $\mathcal{C}_{m-N_1,N_1-N_1'}$, indicating a potential reduction in the overall cost.

We remind that the main purpose of the hybrid-RP is only to replace the first step. It implies that by plugging this hybrid algorithm, the total cost can be computed by

$$\begin{aligned}\mathcal{T}_{Hyb} &= \frac{1}{\mathcal{P}(1)}\left(\mathcal{C}'_{m,N_1,t'} + \frac{1}{\mathcal{P}(2)}\left(\mathcal{C}(2) + \cdots + \frac{1}{\mathcal{P}(k)}\left(\mathcal{C}(k) + \mathcal{S}\right)\right)\right) \\ &= \frac{\mathcal{C}'_{m,N_1,t'}}{\mathcal{P}(1)} + \sum_{j=2}^{k}\frac{\mathcal{C}(j)}{\mathcal{P}'(j)} + \frac{\mathcal{S}}{\mathcal{P}'(k)}.\end{aligned}$$

The full algorithm corresponding to the hybrid-RP is then given by Algorithm 2.

$$\begin{aligned}\mathcal{T}_{Hyb} &= \frac{1}{\mathcal{P}(1)}\left(\mathcal{C}'_{m,N_1,t'} + \frac{1}{\mathcal{P}(2)}\left(\mathcal{C}(2) + \cdots + \frac{1}{\mathcal{P}(k)}\left(\mathcal{C}(k) + \mathcal{S}\right)\right)\right) \\ &= \frac{\mathcal{C}'_{m,N_1,t'}}{\mathcal{P}(1)} + \sum_{j=2}^{k}\frac{\mathcal{C}(j)}{\mathcal{P}'(j)} + \frac{\mathcal{S}}{\mathcal{P}'(k)}.\end{aligned}$$

The full algorithm corresponding to the hybrid-RP is then given by Algorithm 2.

Algorithm 2 Hybrid Reduce and Prange Estimation

Input: LPN parameters (m, n, t)
Output: The total cost of the Hybrid Reduce and Prange algorithm.
for $t_1 = 1$ up to t **do**
  Set $Threshold = \frac{\binom{m}{n}}{\binom{m-t}{n}}$
  Set $Cost = Threshold \cdot n^{2.8}$
  for $\Delta = 1$ up to $n^{2.8}$ **do**
    Compute N_1 such that $\frac{\mathcal{C}'_{m,N_1,t'}}{\mathcal{P}'(1)} < \Delta \cdot Threshold$
    Set $idx = 2$
    while $\exists N_{idx}$ such that $\frac{\mathcal{C}(idx)}{\mathcal{P}'(idx)} < \Delta \cdot Threshold$ **do**
      Set largest N_{idx} such that $\frac{\mathcal{C}(idx)}{\mathcal{P}'(idx)} < \Delta \cdot Threshold$
      Set $idx = idx + 1$
    end while
    Set $N_{idx} = n - \sum_{i=1}^{idx-1} N_i$
    $Cost = \min\left\{Cost, \frac{\mathcal{C}'_{m,N_1,t'}}{\mathcal{P}'(1)} + \sum_{i=2}^{idx-1}\frac{\mathcal{C}(i)}{\mathcal{P}'(i)} + Threshold \cdot N_{idx}^{2.8}\right\}$
  end for
end for
Return $Cost$ and $\{N_i\}_{i=1}^{idx}$

By implementing this hybrid algorithm, a more accurate estimation of security level can be achieved. For detailed implementation results, please refer to the Section 4.

Remark 1. The hybrid-RP could be further optimized by applying the re-using technique from [10] at each iterative step. However, the improvement is expected to be marginal according to our estimations, since the values of N_i for $i \geq 2$ are relatively small. Consequently, reducing the size of such N_i would not substantially improve the estimated results.

3 Regular-RP: Application to regular-LPN

This section provides how to apply the RP to the LPN with regular noise, which is equivalent to the regular syndrome decoding problem. Throughout this section, we set $\mathcal{R} = \mathbb{F}_q$ with $q \geq 2$.

We first introduce a new problem, called the regular syndrome decoding (RSD). RSD was originated [5], and it has a lot of applications in PGC-like protocols [7, 15, 17, 19, 25, 28, 39, 60–62, 67, 71].

Problem 3 (Regular Syndrome Decoding over $\mathbb{F}_q$). Let m, n, t, β be positive integers with $m = t \cdot \beta$ and $\mathbb{F}_q$ be a finite field. Sample a full rank matrix $\mathbf{H} \leftarrow \mathbb{F}_q^{(m-n)\times m}$ and a column vector $\mathbf{e} := (\mathbf{e}_1 \| \mathbf{e}_2 \| \dots \| \mathbf{e}_t) \leftarrow \mathbb{F}_q^m$ such that $\mathbf{e}_i \in \mathbb{F}_2^\beta$ is of Hamming weight $|\mathbf{e}_i| = 1$ for $1 \leq i \leq t$. Given $(\mathbf{H}, \mathbf{y} = \mathbf{H} \cdot \mathbf{e})$, recover the error $\mathbf{e}$.

Note that LPN instances of the form $(\mathbf{A}, \mathbf{b}) = (\mathbf{A}, \mathbf{A} \cdot \mathbf{s} + \mathbf{e})$ can be easily transformed into a problem of finding $\mathbf{e}$ when there exists a matrix $\mathbf{H}$ such that $\mathbf{H} \cdot \mathbf{e} = \mathbf{H} \cdot \mathbf{b}$, and vice versa, as demonstrated in [55, Lemma 4.9]. Therefore, we restrict our attention to the regular-LPN problem.

3.1 Algorithm for regular-LPN over Large Fields

Esser and Satini [37] proposed an algorithm for solving regular-LPN. Among the algorithms, the permutation-based regular ISD algorithm exploits the regular structure to guess the error-free instances. We adopt the technique to our algorithm to tightly estimate the security of regular-LPN over large fields.

To effectively solve regular-LPN over large fields, our strategy focuses on modifying the guessing probability in the RP algorithm. We call it *regular-RP*.

Regular-RP. Specifically, in the initial step, we guess N_1-error free instances to set $\mathbf{e}_{bot} = \mathbf{0} \in \mathbb{Z}_q^{N_1}$. By definition of regular-LPN, it holds that each $\mathbf{e}_i$ has exactly one nonzero element. It immediately implies that the probability of guessing a zero coordinate of $\mathbf{e}_1$ equals to $\frac{\beta-1}{\beta}$. To guess the second zero position, one considers two cases:

- E_1: Find the zero coordinate in $\mathbf{e}_1$.
- E_2: Find the zero coordinate in $\mathbf{e}_i$ for $i \neq 1$.

The probability of event E_1 is $\frac{\beta-2}{\beta-1}$, whereas the probability of event E_2 is $\frac{\beta-1}{\beta}$. Thus, the best strategy is to guess N_1/t-zero coordinates in each $\mathbf{e}_i$. Consequently, the probability of selecting N_1 zero positions is

$$\mathcal{P}^R_{m,t,N_1} = \left(1 - \frac{N_1/t}{\beta}\right)^t = \left(1 - \frac{N_1}{m}\right)^t.$$

Similar to in Section 2.3, we can also calculate the time costs of an iterative version. To this end, we define $\mathcal{P}^R(j), \mathcal{C}(j)$ and $\mathcal{S}^R$ as follows:

$$\begin{aligned}
\mathcal{P}^R(j) &= \mathcal{P}^R_{(m-\sum_{i=1}^{j-1} N_i),t,N_j} \\
\mathcal{C}(j) &= \mathcal{C}_{(m-\sum_{i=1}^{j-1} N_i),N_j} \\
\mathcal{S}^R &= \frac{1}{\mathcal{P}^R_{(m-\sum_{i=1}^{k} N_i),t,(n-\sum_{i=1}^{k} N_i)}} \cdot \left(n - \sum_{i=1}^{k} N_i\right)^\omega
\end{aligned}$$

Then, under the notation, the total cost of level-k regular-RP algorithm, denoted by $\mathcal{T}^R$, can be expressed as

$$\frac{1}{\mathcal{P}^R(1)}\left(\mathcal{C}(1) + \frac{1}{\mathcal{P}^R(2)}\left(\mathcal{C}(2) + \cdots + \frac{1}{\mathcal{P}^R(k)}\left(\mathcal{C}(k) + \mathcal{S}^R\right)\right)\right)$$

To simplify it, we also define $\mathcal{P}'^R(j)$ as $\mathcal{P}'^R(j) = \prod_{i=1}^{j} \mathcal{P}^R(i)$. We then denote the total cost as

$$\mathcal{T}^R = \sum_{j=1}^{k} \frac{\mathcal{C}(j)}{\mathcal{P}'^R(j)} + \frac{\mathcal{S}^R}{\mathcal{P}'^R(k)}.$$

Hybrid-regular-RP. As in Section 2.4, we can more optimize the attack via hybrid approach. We recall that, in the hybrid-RP, the $\mathbf{e}_{bot}$ contains N_1' zero coordinates with probability $\mathcal{P}'_{N_1,N_1',m,t}$ for the standard LPN problem after the first replacement. It suffices to compute such a probability depending on regular-LPN.

In the case of regular-LPN, we first remark that $\mathbf{e}_{bot}$ is still a regular noise because of our guessing strategy E_2. Let $\mathbf{e}_{bot} = (\mathbf{e}_1'\|\mathbf{e}_2'\|\ldots\|\mathbf{e}_t') \in \mathbb{F}_q^{N_1}$, where $\mathbf{e}_i' \in \mathbb{F}_q^{N_1/t}$ is a sub-vector of $\mathbf{e}_i$. Similar to the analysis in Section 2.4, we assume that $\mathbf{e}_{bot}$ has N_1' zero coordinates. Since $\mathbf{e}_i'$ has precisely one nonzero entry for each $i \in [t]$ and $\mathbf{e}_{bot}$ contains exactly $N_1 - N_1'$ nonzero coordinates, it satisfies that there is a subset $S \subset \{1, 2, \ldots, t\}$ of size $t - (N_1 - N_1')$ such that $\mathbf{e}_j'$ is the zero vector for every $j \in S$. We further note that the probability of each $\mathbf{e}_i'$ has nonzero coordinates with probability $\frac{N_1}{\beta}$.

Combining all the facts together, $\mathbf{e}_{bot}$ contains N_1' zero coordinates with probability

$$\mathcal{P}'^R_{N_1,N_1',m,t} = \binom{t}{N_1 - N_1'} \cdot \left(\frac{N_1}{\beta}\right)^{N_1-N_1'} \cdot \left(1 - \frac{N_1}{\beta}\right)^{t-(N_1-N_1')}.$$

By plugging it into the Equation (1), one can compute the time complexity of the hybrid regular-RP:

$$\mathcal{T}^R_{Hyb} = \frac{\mathcal{C}'_{m,N_1,t'}}{\mathcal{P}^R(1)} + \sum_{j=2}^{k} \frac{\mathcal{C}(j)}{\mathcal{P}'^R(j)} + \frac{\mathcal{S}^R}{\mathcal{P}'^R(k)}.$$

3.2 Algorithm for regular-LPN over binary field

According to [18], the regularity of an error distribution τ provides an extra relations: For any $1 \leq i \leq t$, we observe that

$$1 - \sum_{j=1}^{\beta} e_{i,j} = 0,$$

where $e_{i,j}$ is the j-th entry of a vector $\mathbf{e}_i$. Consequently, the attack can easily get t linear equations from the regularity of $\tau_{m,t}$.

We can parse $\mathbf{b} = (\mathbf{b}_1 \| \cdots \| \mathbf{b}_t)$ where $\mathbf{b}_i \in \mathbb{F}_2^{\beta}$, and our attack exploits this fact as follows: For every $1 \leq i \leq t$,

$$\begin{aligned} \sum_{j=1}^{\beta} b_{i,j} &= (\sum_{j=1}^{\beta} \mathbf{a}_j^T) \cdot \mathbf{s} + \sum_{j=1}^{\beta} e_{i,j} \bmod 2 \\ &= (\sum_{j=1}^{\beta} \mathbf{a}_j^T) \cdot \mathbf{s} + 1 \bmod 2 \end{aligned}$$

Here, $b_{i,j}$ is the j-th entry of $\mathbf{b}_i$, and $\mathbf{a}_j^T$ is the j-th row vector of $\mathbf{A}$. Thus, we automatically obtain an error-free polynomial g_i of the form

$$g_i(\mathbf{x}) = \sum_{j=1}^{\beta} b_{i,j} - 1 - (\sum_{j=1}^{\beta} \mathbf{a}_j^T) \cdot \mathbf{x}.$$

This exploitation can significantly enhance our attack. Specifically, the purpose of RP is to collect error-free polynomials, and the t error-free polynomials obtained from the regularity of $\tau_{m,t}$ reduce the number of error-free polynomials that we need to obtain. Consequently, we need only $n - t$ error-free polynomials to solve regular-LPN.

The remaining part is to obtain $n - t$ error-free polynomial through RP in Section 3.1.

4 Concrete Estimation of LPN via Reduce and Prange algorithm

In this section, we provide the estimated results of our algorithms for solving the LPN over $\mathcal{R}$. The results are given by

– Table 1: Numerical results of RP and hybrid-RP for solving LPN over $\mathbb{F}_{2^{128}}$
– Table 2: Numerical results of RP for solving LPN over various rings
– Table 3: Numerical results of regular-RP for solving regular-LPN

We further compare the results with recent attacks [18,37,49] depending on problems in each tables. Throughout this section, time complexity means the number of arithmetic operations in $\mathcal{R}$. For this purpose, we first recall the parameters of RP and its variants for LPN over finite fields.

– q: the size of fields $\approx 2^{128}$.
– m: the number of LPN instances.
– n: the dimension of LPN secret $\mathbf{s}$.
– t: the number of nonzero entries in $\mathbf{e}$.

Our focus is mainly on the parameter regime $(m, q, t) = (poly(n), n^{w(1)}, o(m))$, which has been widely used in several cryptographic applications including those presented in [7,8,14,16,17,25,29,62,67,71], and which is affected by our analysis of the cost of solving LPN over finite fields. Specifically, our comparison is limited to ISD and its variants, which were highlighted for their substantial impact in the recent work by Liu et al. [49]. Therefore, these serve as the most relevant benchmarks for evaluating our results. [2]

For fair comparisons, we adopt the cost models of prior attacks, using the notations in earlier sections. The majority of these cost models are in the recent analysis conducted by Liu et al. [49]. We remind comments in [49]:

– For the parameter regime $\mathcal{R} = \mathbb{F}_q$ with prime power $q = n^{w(1)}$, $t = o(m)$ and $(1 + \beta) \cdot n \leq m = poly(n)$ for some constant β, the statistical decoding including the 2.0 variant [23] needs more cost than Prange's algorithm [59].
– For the parameter setup, information set decoding and its variants outperforms than other algorithms.

It is important to note that many algorithms, including ours, utilize Gaussian elimination as a subroutine algorithm, which has an asymptotic runtime of $O(k^\omega)$ for a matrix rank k and linear algebra constant ω. Therefore, it is necessary to eliminate the hidden constant in the big-O notation to accurately measure the concrete security estimation for LPN over large fields.

Our estimations are based on the same cost model of the time complexity of the Gaussian elimination algorithm on a $k \times k$ matrix. We utilize a cost model of $k^{2.8}$ as in the previous works [14, 49].[3]

Prange's Algorithm. As described in Section 2.1, the basic idea behind our algorithm is to collect several error-free LPN instances (*i.e.*, $e_i = 0$) and use Gaussian elimination to solve the corresponding linear system. Specifically, given an $n \times n$ matrix $\mathbf{A}'$ and $\mathbf{b}'$ such that $\mathbf{A}' \cdot \mathbf{s} = \mathbf{b}'$, we can recover the secret vector $\mathbf{s}$ by computing the inverse matrix of $\mathbf{A}'$.

[2] Other algorithms for solving LPN are given in Appendix A.
[3] The exponent is inspired by the Strassen matrix multiplication algorithm.

The probability of obtaining such a matrix $\mathbf{A}'$ is $\frac{\binom{m-n}{t}}{\binom{m}{t}}$, and the cost of computing the inverse of $\mathbf{A}'$ is $n^{2.8}$. Consequently, the cost is $n^{2.8} \cdot \left(\frac{\binom{m}{t}}{\binom{m-n}{t}}\right)$.

Furthermore, it is worth noting that LPN instances $(\mathbf{A}, \mathbf{b}) = (\mathbf{A}, \mathbf{A} \cdot \mathbf{s} + \mathbf{e})$ can be easily converted into a problem of finding $\mathbf{e}$ when $\mathbf{H} \cdot \mathbf{e} = \mathbf{H} \cdot \mathbf{b}$ for some matrix $\mathbf{H}$, and vice versa, as shown in [55, Lemma 4.9]. In this case, the cost is $(m-n)^{2.8} \cdot \left(\frac{\binom{m}{t}}{\binom{m-n}{t}}\right)$.

Consequently, the Prange's cost model is defined as follows:

$$\texttt{Prange\_cost\_model} = \log\left(\min\{n^{2.8}, (m-n)^{2.8}\} \cdot \left(\frac{\binom{m}{t}}{\binom{m-n}{t}}\right)\right)$$

Information Set Decoding. In accordance with the findings reported in [49], the cost of information set decoding (ISD) attacks is shown to be comparable to that of Prange's algorithm, provided that q is sufficiently large. With this in mind, we provide a summary of the computational costs associated with ISD and its variants in this section. We take into account several remarks provided in [49] to ensure a fair comparison. Furthermore, we also consider the estimations presented in [49] on the hardness of LPN over $\mathbb{F}_q$.

Moreover, the work in [49] is primarily concerned with the hardness of LPN over $\mathbb{F}_q$ for $q \geq 256$, as the authors argue that other cases are less relevant to applications such as PCG, MPC, and ZK. The estimator for low-noise LPN presented in [49] is a generalization of the SD-ISD algorithm by Peters [58]. However, the paper does not cover other LPN-solving algorithms, such as those in [6, 13, 34–36, 41, 43, 52, 54, 63], which have a hidden polynomial overhead that makes them less efficient to analyze in terms of concrete cost or space consumption. Furthermore, it is currently unclear how the techniques proposed in [9, 51] can be extended to solve LPN over $\mathbb{F}_q$ for large values of q. For a detailed discussion of this issue, we refer the reader to the original paper [49].

In summary, [49] estimates the hardness of LPN over $\mathbb{F}_q$ through SD-ISD attack.

Lemma 4.1 (Cost of SD-ISD attack [49]) *The (m, n, t)-LPN problem over $\mathbb{F}_q$ can be solved by the SD-ISD variant in expected time*

$$T_{SD}(m, n, t) = \min_{p,\ell} \frac{1}{\mathcal{P}(p, \ell)} \left(T_{Gauss} + 2L \cdot m \left(1 + \frac{L}{q^\ell}\right)\right)$$

where $T_{Gauss} = \frac{(m-n-\ell)\cdot m^2}{\log(m-n-\ell)}$, $L = \binom{\frac{n+\ell}{2}}{p/2} \cdot (q-1)^{p/2}$ *and* $\mathcal{P}(p, \ell) = \frac{\binom{m-n-\ell}{t-p}}{\binom{m}{t}} \cdot \frac{L^2}{(q-1)^p}$.[4]

[4] As the paper [49] only presents an estimation of LPN over $\mathbb{F}_2$, we rely on the formulation provided by their Python script to describe our approach.

4.1 Numerical Results of LPN (over large fields)

This section presents the numerical results obtained from our attack estimation for different parameter settings, especially PCG-like protocol setting. We developed a SageMath [64] script to search for near-optimal parameterization for our attack. The original algorithm would have examined the attack complexity that is minimized for all Δ values, but we examined the values via the binary search with depth ≤ 12 to find an approximate minimum quickly with a constraint $\Delta \leq n^{2.8}$. The cost we found also provides a sufficiently small value compared to the original cost.

Our analysis indicates that for the parameters used in [49], the bit-security of LPN over large fields is reduced by 5-11 bits when $\log q = 128$. Note that in the case of LPN over $\mathbb{F}_2$, the SD-ISD attack proposed in [49] outperforms our algorithm. Similarly, for solving LPN over $\mathbb{Z}_{2^{128}}$ and $\mathbb{Z}_4$, the SD-ISD attack proposed in [49] outperforms our algorithm. This is because LPN over these rings can be reduced to LPN over $\mathbb{F}_2$ with fewer Hamming weights, as reported in [49]. Indeed, as noted in [49], we estimate the bit-security of (m, n, t)-LPN over $\mathbb{Z}_{2^\lambda}$ with $\lambda \geq 2$ as that of $(m, n, \frac{2^{\lambda-1}}{2^\lambda-1} \cdot t)$-LPN over $\mathbb{F}_2$.

We also provide comparison results between [49] and ours for a few parameters. The results are given by Table 1, Table 2.

Table 1. Comparison results with previous results [49] and ours. All values are logarithmic scale of arithmetic operations over $\mathbb{F}_q$ with the field size $\log q = 128$. In our works, 'RP' column presents the estimated result of Reduce and Prange algorithm. The last column displays the numerical results of a hybrid-RP. For the dash(-) in the hybrid column, we terminated the experiment because it was taking too much time.

Parameters			[49][5]		**This work**	
m	n	t	Gauss	ISD	RP	Hybrid
2^{10}	652	57	111	111	**102**	**101**
2^{12}	1589	98	100	100	**91**	**89**
2^{14}	3482	198	101	101	**94**	**93**
2^{16}	7391	389	103	103	**96**	**95**
2^{18}	15536	760	105	105	**101**	105
2^{20}	32771	1419	107	107	**102**	-
2^{22}	67440	2735	108	108	**104**	-
2^{12}	3072	44	117	117	109	106
2^{14}	12288	39	111	111	105	101
2^{16}	49152	34	107	107	102	-
2^{18}	196608	32	108	108	105	-

Table 2. Comparison results with previous results [49] for various rings. The numerical results presented in this table indicate the result of RP, not the hybrid-RP. All values are logarithmic scale of the number of arithmetic operations over $\mathcal{R}$.[7]

Parameters			[49]					**This work**				
m	n	t	$\mathbb{F}_{2^{128}}$	$\mathbb{F}_{2^8}$	$\mathbb{Z}_{2^{128}}$	$\mathbb{Z}_4$	$\mathbb{F}_2$	$\mathbb{F}_{2^{128}}$	$\mathbb{F}_{2^8}$	$\mathbb{Z}_{2^{128}}$	$\mathbb{Z}_4$	$\mathbb{F}_2$
2^{10}	652	106	194	186	89	116	176	183	183	95	124	183
2^{12}	1589	172	155	146	76	95	131	144	144	83	103	144
2^{14}	3482	338	150	144	78	95	132	141	141	84	103	141
2^{16}	7391	667	151	148	82	99	135	143	143	87	106	143
2^{18}	15336	1312	153	153	87	104	139	145	145	90	108	145
2^{20}	32771	2467	155	157	92	108	143	147	147	94	111	147
2^{22}	67440	4788	156	160	97	113	147	150	150	97	114	150

Remark 2. Some readers might be curious about the configuration of the sub-dimension sets $\{N_i\}_{i=1}^{idx}$. Given our findings, with idx typically ranging from over 20 to as high as 2000, accurately defining the set can be challenging. Concurrently, it has been noted that the size of N_1 often exceeds $N/3$. Similar occurrences are also observed in the context of the regular-LPN.

4.2 Numerical Results of regular-LPN

This section provides numerical results for solving regular-LPN for various parameters, and gives comparison results with [18, 37, 49]. Here, as in the previous section, we mainly cover regular-LPN parameters mainly used in the PCG setting. Thus, we do not cover parameters in [22].

We first note that [18, 49] provides numerical results of the bit-security of regular-LPN over $\mathbb{F}_q$ with $\log q = 128$ since both algorithms rarely depend on the field size. Conversely, [37] solely provides the bit-security of regular-LPN over $\mathbb{F}_2$. Thus, we revisit the algorithms in [37] and re-estimate the bit-security of regular-LPN over $\mathbb{F}_{2^{128}}$.

[37] proposes three types of algorithms: permutation-based regular-ISD (P-ISD), enumeration-based regular-ISD (E-ISD), and representation-based regular-ISD (R-ISD), each adapting well-known algorithms [9, 51, 59] for the specific task of solving regular-LPN. Among these, we specifically focus on the P-ISD algorithm. This is because the other algorithms require the collection of a large number of collision vectors over $\mathbb{Z}_q$, which significantly increase the concrete

[5] [49] also presents numerical results for the statistical decoding 2.0 attack [24]. However, we did not take these results into account because they showed lower performance compared to than Gauss and ISD results in these parameters in the table.

[7] The latest version of [49] at ePrint is 20231008:131837. The results these parameters of parameters are in 20230218:110851 of [49].

complexity. Indeed, numerical results from [49, Tab 3. Tab. 4] indicate that the Gauss costs are always comparable to ISD costs for (regular-)LPN over $\mathbb{F}_{2^{128}}$ for all PCG parameter setting, although ISD consistently surpasses the Gauss in terms of cost for (regular-)LPN over $\mathbb{F}_2$.

Revisiting P-ISD for regular-LPN over $\mathbb{F}_q$ with $q \gg 2$. As discussed in Section 3.1 and [37, Sec. 4.1], we can just compute the time cost as follows

$$\left(1 - \frac{\binom{n}{t}}{\binom{m}{t}}\right)^{-t} \cdot \min\{n^{2.8}, (m-n)^{2.8}\} = \left(1 - \frac{n}{m}\right)^{-t} \cdot \min\{n^{2.8}, (m-n)^{2.8}\}$$

for solving (m, n, t)-regular-LPN. We further note that in [37, Sec. 4.1], they computed the time complexity

$$\left(1 - \frac{n-t}{m}\right)^{-t} \cdot \min\{n^{2.8}, (m-n)^{2.8}\}$$

because one can directly obtain extra t error-free polynomials as in Section 3.2. However, in case of regular-LPN over $\mathbb{F}_q$, it does not occur. Hence, we re-estimate the bit-security of regular LPN over $\mathbb{F}_{2^{128}}$ as follows:

$$\left(1 - \frac{n}{m}\right)^{-t} \cdot \min\{n^{2.8}, (m-n)^{2.8}\}.$$

For a binary case, we wrote down the best performance used in [37].

To measure the cost of the regular-LPN over $\mathbb{F}_{2^{128}}$, we use an algorithm in Section 3.1. For the regular-LPN with binary fields, we can employ a technique in Section 3.2. Thus, the regular noise with binary fields additionally provides t error-free polynomials. By augmenting the extra error-free instances, the regular-LPN can be interpreted as $(m+t, n, t)$-regular-LPN samples. We then apply the algorithm Section 3.1 to get $(m, n-t, t)$-regular-LPN samples for the augmented error-free equations. Next, by applying the whole algorithm in Section 3.1 on the $(m, n-t, t)$-regular-LPN samples, one can solve the regular-LPN over $\mathbb{F}_2$.

The numerical results are reported in Table 3, which demonstrates that our attack outperforms several parameter settings. To estimate the cost, we use a hybrid-regular-RP algorithm that combines a regular-RP with [10] as in the hybrid-RP.

References

1. M. Alekhnovich. More on average case vs approximation complexity. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 298–307. IEEE, 2003.
2. B. Applebaum, B. Barak, and A. Wigderson. Public-key cryptography from different assumptions. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 171–180, 2010.

Table 3. Numerical results for regular-LPN over $\mathbb{F}_2$ and $\mathbb{F}_{2^{128}}$. The results from [49] actually solves the complexity of LPN over $\mathbb{F}_{2^{128}}$.All values are logarithmic scale of the number of the number of arithmetic operations over $\mathbb{F}_{2^{128}}$.

Parameters			[49]		[18]		[37]		**This work**	
m	n	t	$\mathbb{F}_{2^{128}}$	$\mathbb{F}_2$	$\mathbb{F}_{2^{128}}$	$\mathbb{F}_2$	$\mathbb{F}_{2^{128}}$	$\mathbb{F}_2$	$\mathbb{F}_{2^{128}}$	$\mathbb{F}_2$
2^{10}	652	106	194	159	179	145	178	113	**150**	117
2^{12}	1589	172	155	121	150	135	151	109	**136**	120
2^{14}	3482	338	150	122	150	138	149	118	**137**	125
2^{16}	7391	667	151	125	150	139	151	126	**141**	130
2^{18}	15336	1312	153	129	**133**	122	154	126	147	137
2^{20}	32771	2467	155	135	**131**	125	155	138	148	139
2^{22}	67440	4788	152	135	**110**	103	156	140	151	143
2^{10}	652	57	111	90	111	101	107	76	**94**	84
2^{12}	1589	98	100	80	107	103	99	78	**89**	84
2^{14}	3482	198	101	83	110	106	101	84	**93**	88
2^{16}	7391	389	103	87	111	108	103	90	**97**	92
2^{18}	15336	760	105	92	107	104	105	93	**100**	97
2^{20}	32771	1419	107	97	**102**	98	107	98	103	100
2^{22}	67440	2735	108	99	**104**	103	108	103	105	102

3. B. Applebaum, D. Cash, C. Peikert, and A. Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, volume 5677 of *Lecture Notes in Computer Science*, pages 595–618. Springer, 2009.
4. B. Applebaum, I. Damgård, Y. Ishai, M. Nielsen, and L. Zichron. Secure arithmetic computation with constant computational overhead. In *Annual International Cryptology Conference*, pages 223–254. Springer, 2017.
5. D. Augot, M. Finiasz, and N. Sendrier. A family of fast syndrome based cryptographic hash functions. In *Mycrypt*, volume 3715, pages 64–83. Springer, 2005.
6. M. Baldi, A. Barenghi, F. Chiaraluce, G. Pelosi, and P. Santini. A finite regime analysis of information set decoding algorithms. *Algorithms*, 12(10):209, 2019.
7. C. Baum, L. Braun, A. Munch-Hansen, and P. Scholl. Moz$\mathbb{Z}_2^k$arella: efficient vector-ole and zero-knowledge proofs over z 2 k. In *Advances in Cryptology–CRYPTO 2022: 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15–18, 2022, Proceedings, Part IV*, pages 329–358. Springer, 2022.
8. C. Baum, A. J. Malozemoff, M. B. Rosen, and P. Scholl. Mac'n'cheese mac′ n′ cheese: Zero-knowledge proofs for boolean and arithmetic circuits with nested disjunctions. In *Advances in Cryptology–CRYPTO 2021: 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16–20, 2021, Proceedings, Part IV 41*, pages 92–122. Springer, 2021.
9. A. Becker, A. Joux, A. May, and A. Meurer. Decoding random binary linear codes in 2 n/20: How 1+ 1= 0 improves information set decoding. In *Annual international conference on the theory and applications of cryptographic techniques*, pages 520–536. Springer, 2012.
10. D. J. Bernstein, T. Lange, and C. Peters. Attacking and defending the mceliece cryptosystem. In J. Buchmann and J. Ding, editors, *Post-Quantum Cryptography*, pages 31–46, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.
11. A. Blum, M. L. Furst, M. J. Kearns, and R. J. Lipton. Cryptographic primitives based on hard learning problems. In *Advances in Cryptology - CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings*, volume 773 of *Lecture Notes in Computer Science*, pages 278–291. Springer, 1993.
12. A. Blum, A. Kalai, and H. Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *Journal of the ACM (JACM)*, 50(4):506–519, 2003.
13. L. Both and A. May. Decoding linear codes with high error rate and its impact for lpn security. In *International Conference on Post-Quantum Cryptography*, pages 25–46. Springer, 2018.
14. E. Boyle, G. Couteau, N. Gilboa, and Y. Ishai. Compressing vector OLE. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*, pages 896–912. ACM, 2018.
15. E. Boyle, G. Couteau, N. Gilboa, Y. Ishai, L. Kohl, N. Resch, and P. Scholl. Correlated pseudorandomness from expand-accumulate codes. In *Advances in Cryptology–CRYPTO 2022: 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15–18, 2022, Proceedings, Part II*, pages 603–633. Springer, 2022.
16. E. Boyle, G. Couteau, N. Gilboa, Y. Ishai, L. Kohl, P. Rindal, and P. Scholl. Efficient two-round ot extension and silent non-interactive secure computation. In

Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, pages 291–308, 2019.
17. E. Boyle, G. Couteau, N. Gilboa, Y. Ishai, L. Kohl, and P. Scholl. Efficient pseudorandom correlation generators: Silent ot extension and more. In *Advances in Cryptology–CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2019, Proceedings, Part III 39*, pages 489–518. Springer, 2019.
18. P. Briaud and M. Øygarden. A new algebraic approach to the regular syndrome decoding problem and implications for pcg constructions. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 391–422. Springer, 2023.
19. D. Bui and G. Couteau. Private set intersection from pseudorandom correlation generators. *IACR Cryptol. ePrint Arch.*, 2022:334, 2022.
20. A. Canteaut and F. Chabaud. A new algorithm for finding minimum-weight words in a linear code: Application to mceliece's cryptosystem and to narrow-sense bch codes of length 511. *IEEE Transactions on Information Theory*, 44(1):367–378, 1998.
21. R. Canto Torres and N. Sendrier. Analysis of information set decoding for a sublinear error weight. In *International Workshop on Post-Quantum Cryptography*, pages 144–161. Springer, 2016.
22. E. Carozza, G. Couteau, and A. Joux. Short signatures from regular syndrome decoding in the head. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 532–563. Springer, 2023.
23. K. Carrier, T. Debris-Alazard, C. Meyer-Hilfiger, and J.-P. Tillich. Statistical decoding 2.0: Reducing decoding to lpn. In *Advances in Cryptology–ASIACRYPT 2022: 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5–9, 2022, Proceedings, Part IV*, pages 477–507. Springer, 2022.
24. K. Carrier, T. Debris-Alazard, C. Meyer-Hilfiger, and J.-P. Tillich. Statistical decoding 2.0: Reducing decoding to lpn. *Asiacrpyt 2022*, 2022.
25. G. Couteau, P. Rindal, and S. Raghuraman. Silver: silent vole and oblivious transfer from hardness of decoding structured ldpc codes. In *Advances in Cryptology–CRYPTO 2021: 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16–20, 2021, Proceedings, Part III*, pages 502–534. Springer, 2021.
26. I. Damgård and S. Park. Is public-key encryption based on LPN practical? *IACR Cryptol. ePrint Arch.*, 2012:699, 2012.
27. I. Damgård, V. Pastro, N. Smart, and S. Zakarias. Multiparty computation from somewhat homomorphic encryption. In *Advances in Cryptology–CRYPTO 2012: 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, pages 643–662. Springer, 2012.
28. S. Dittmer, Y. Ishai, S. Lu, and R. Ostrovsky. Authenticated garbling from simple correlations. In *Advances in Cryptology–CRYPTO 2022: 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15–18, 2022, Proceedings, Part IV*, pages 57–87. Springer, 2022.
29. S. Dittmer, Y. Ishai, S. Lu, and R. Ostrovsky. Improving line-point zero knowledge: Two multiplications for the price of one. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 829–841, 2022.
30. S. Dittmer, Y. Ishai, S. Lu, and R. Ostrovsky. Improving line-point zero knowledge: Two multiplications for the price of one. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 829–841, 2022.

31. S. Dittmer, Y. Ishai, and R. Ostrovsky. Line-point zero knowledge and its applications. *Cryptology ePrint Archive*, 2020.
32. Y. Dodis, E. Kiltz, K. Pietrzak, and D. Wichs. Message authentication, revisited. In D. Pointcheval and T. Johansson, editors, *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*, volume 7237 of *Lecture Notes in Computer Science*, pages 355–374. Springer, 2012.
33. N. Döttling, S. Ghosh, J. B. Nielsen, T. Nilges, and R. Trifiletti. Tinyole: Efficient actively secure two-party computation from oblivious linear function evaluation. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS*, page 2263–2276, New York, NY, USA, 2017. Association for Computing Machinery.
34. I. Dumer. On minimum distance decoding of linear codes. In *Proc. 5th Joint Soviet-Swedish Int. Workshop Inform. Theory*, pages 50–52, 1991.
35. A. Esser and E. Bellini. Syndrome decoding estimator. In *Public-Key Cryptography–PKC 2022: 25th IACR International Conference on Practice and Theory of Public-Key Cryptography, Virtual Event, March 8–11, 2022, Proceedings, Part I*, pages 112–141. Springer, 2022.
36. A. Esser, A. May, and F. Zweydinger. Mceliece needs a break–solving mceliece-1284 and quasi-cyclic-2918 with modern isd. In *Advances in Cryptology–EUROCRYPT 2022: 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30–June 3, 2022, Proceedings, Part III*, pages 433–457. Springer, 2022.
37. A. Esser and P. Santini. Not just regular decoding: Asymptotics and improvements of regular syndrome decoding attacks. *Cryptology ePrint Archive*, 2023.
38. J.-B. Fischer and J. Stern. An efficient pseudo-random generator provably as secure as syndrome decoding. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 245–255. Springer, 1996.
39. N. Franzese, J. Katz, S. Lu, R. Ostrovsky, X. Wang, and C. Weng. Constant-overhead zero-knowledge for ram programs. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 178–191, 2021.
40. S. Ghosh, J. B. Nielsen, and T. Nilges. Maliciously secure oblivious linear function evaluation with constant overhead. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 629–659. Springer, 2017.
41. C. T. Gueye, J. B. Klamti, and S. Hirose. Generalization of bjmm-isd using may-ozerov nearest neighbor algorithm over an arbitrary finite field. In *Codes, Cryptology and Information Security: Second International Conference, C2SI 2017, Rabat, Morocco, April 10–12, 2017, Proceedings-In Honor of Claude Carlet*, pages 96–109. Springer, 2017.
42. C. Hazay, P. Scholl, and E. Soria-Vazquez. Low cost constant round mpc combining bmr and oblivious transfer. *Journal of Cryptology*, 33(4):1732–1786, 2020.
43. S. Hirose. May-ozerov algorithm for nearest-neighbor problem over and its application to information set decoding. In *Innovative Security Solutions for Information Technology and Communications: 9th International Conference, SECITC 2016, Bucharest, Romania, June 9-10, 2016, Revised Selected Papers*, pages 115–126. Springer, 2016.
44. Y. Ishai, M. Prabhakaran, and A. Sahai. Secure arithmetic computation with no honest majority. In *Theory of Cryptography Conference*, pages 294–314. Springer, 2009.

45. A. Jain, S. Krenn, K. Pietrzak, and A. Tentes. Commitments and efficient zero-knowledge proofs from learning parity with noise. In *Advances in Cryptology - ASIACRYPT 2012 - 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6, 2012. Proceedings*, volume 7658 of *Lecture Notes in Computer Science*, pages 663–680. Springer, 2012.
46. A. Jain, H. Lin, and A. Sahai. Indistinguishability obfuscation from well-founded assumptions. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 60–73, 2021.
47. M. Keller, E. Orsini, and P. Scholl. Mascot: faster malicious arithmetic secure computation with oblivious transfer. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 830–842, 2016.
48. E. Kiltz, K. Pietrzak, D. Venturi, D. Cash, and A. Jain. Efficient authentication from hard learning problems. *J. Cryptol.*, 30(4):1238–1275, 2017.
49. H. Liu, X. Wang, K. Yang, and Y. Yu. The hardness of lpn over any integer ring and field for pcg applications. To appear at Eurocrypt'24.
50. V. Lyubashevsky and D. Masny. Man-in-the-middle secure authentication schemes from LPN and weak prfs. In *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 308–325. Springer, 2013.
51. A. May, A. Meurer, and E. Thomae. Decoding random linear codes in $\tilde{O}(2^{0.054n})$. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 107–124. Springer, 2011.
52. A. May and I. Ozerov. On computing nearest neighbors with applications to decoding of binary linear codes. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 203–228. Springer, 2015.
53. A. Meurer. *A coding-theoretic approach to cryptanalysis*. PhD thesis, Verlag nicht ermittelbar, 2013.
54. A. Meurer. *A coding-theoretic approach to cryptanalysis*. PhD thesis, Verlag nicht ermittelbar, 2013.
55. D. Micciancio and P. Mol. Pseudorandom knapsacks and the sample complexity of lwe search-to-decision reductions. In *Advances in Cryptology–CRYPTO 2011: 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings 31*, pages 465–484. Springer, 2011.
56. M. Naor and B. Pinkas. Oblivious polynomial evaluation. *SIAM Journal on Computing*, 35(5):1254–1281, 2006.
57. J. B. Nielsen, P. S. Nordholt, C. Orlandi, and S. S. Burra. A new approach to practical active-secure two-party computation. In *Advances in Cryptology–CRYPTO 2012: 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, pages 681–700. Springer, 2012.
58. C. Peters. Information-set decoding for linear codes over $\mathfrak{U}_q$. In *International Workshop on Post-Quantum Cryptography*, pages 81–94. Springer, 2010.
59. E. Prange. The use of information sets in decoding cyclic codes. *IRE Transactions on Information Theory*, 8(5):5–9, 1962.
60. S. Raghuraman and P. Rindal. Blazing fast psi from improved okvs and subfield vole. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 2505–2517, 2022.
61. P. Rindal and P. Schoppmann. Vole-psi: fast oprf and circuit-psi from vector-ole. In *Advances in Cryptology–EUROCRYPT 2021: 40th Annual International*

Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17–21, 2021, Proceedings, Part II, pages 901–930. Springer, 2021.
62. P. Schoppmann, A. Gascón, L. Reichert, and M. Raykova. Distributed vector-ole: improved constructions and implementation. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 1055–1072, 2019.
63. J. Stern. A method for finding codewords of small weight. In *International Colloquium on Coding Theory and Applications*, pages 106–113. Springer, 1988.
64. The Sage Developers. *SageMath, the Sage Mathematics Software System (Version x.y.z)*, YYYY. `https://www.sagemath.org`.
65. X. Wang, S. Ranellucci, and J. Katz. Authenticated garbling and efficient maliciously secure two-party computation. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pages 21–37, 2017.
66. X. Wang, S. Ranellucci, and J. Katz. Global-scale secure multiparty computation. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 39–56, 2017.
67. C. Weng, K. Yang, J. Katz, and X. Wang. Wolverine: fast, scalable, and communication-efficient zero-knowledge proofs for boolean and arithmetic circuits. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 1074–1091. IEEE, 2021.
68. C. Weng, K. Yang, Z. Yang, X. Xie, and X. Wang. Antman: Interactive zero-knowledge proofs with sublinear communication. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 2901–2914, 2022.
69. K. Yang, P. Sarkar, C. Weng, and X. Wang. Quicksilver: Efficient and affordable zero-knowledge proofs for circuits and polynomials over any field. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 2986–3001, 2021.
70. K. Yang, X. Wang, and J. Zhang. More efficient mpc from improved triple generation and authenticated garbling. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 1627–1646, 2020.
71. K. Yang, C. Weng, X. Lan, J. Zhang, and X. Wang. Ferret: Fast extension for correlated ot with small communication. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 1607–1626, 2020.
72. Y. Yu and J. P. Steinberger. Pseudorandom functions in almost constant depth from low-noise LPN. In *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 154–183. Springer, 2016.
73. Y. Yu and J. Zhang. Cryptography with auxiliary input and trapdoor from constant-noise LPN. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I*, volume 9814 of *Lecture Notes in Computer Science*, pages 214–243. Springer, 2016.

A Other algorithms for Solving LPN over Large Fields

In this section, we provide a brief overview of prior works, such as the Blum-Kalai-Wasserman (BKW) algorithm and statistical decoding algorithms, that

are commonly used for solving a standard LPN problem. We then proceed to show that these algorithms are not competitive when applied to LPN over large fields.

BKW framework. Blum *et al.* [12] proposed an algorithm for a standard LPN. This approach (intuitively) aims at finding LPN instances $(\mathbf{a}_i, \mathbf{b}_i)$ such that the first-k coordinates of $\mathbf{a}_i$ are identical. This case happens with $2^{O(k)}$-samples.

From the set, one can have new LPN instances, $(\mathbf{a}_i - \mathbf{a}_j, \mathbf{b}_i - \mathbf{b}_j)$, of dimension $n - k$. Iterating the above process, adversary eventually intends to get a LPN instance of small dimension and it (may) gives a corresponding coordinate of secret $\mathbf{s}$.

From the high-level description, it can be inferred that $q^{O(k)}$ samples are required to get LPN pairs of k coordinates colliding. That is the whole complexity of BKW heavily depends on the size of underlying space q.

Statistical Decoding framework. While the BKW and ISD algorithms aims at recover the secret element $\mathbf{s}$ directly, this low weight parity check (LPC) algorithm intends to distinguish between random instances and LPN instances. In other words, this algorithm can determine $\mathbf{b}$ equals to $\mathbf{A} \cdot \mathbf{s} + \mathbf{e} \bmod 2$ or a random element.

For the purpose, the main task of this algorithm is to find a low-hamming weight matrix $\mathbf{H}$, which holds that $\mathbf{H} \cdot \mathbf{A} = \mathbf{0}$. It then leads to give a new term $\mathbf{H} \cdot \mathbf{b} \bmod 2$. While the term has a uniform distribution for a random vector, it has bias for a LPN sample $\mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$ since this term is identical to a product of two low hamming weight terms $\mathbf{H} \cdot \mathbf{e}$. Repeating this algorithm, one can deduce the distribution $\mathbf{b}$.

On the other hand, Blum *et al.* offered a search to decision reduction [11,38]. The reduction outline consists of two steps: The first step is to sample a random vector $\mathbf{s}'$ and reconstruct a new sample $(\mathbf{A}, \mathbf{b} + \mathbf{A} \cdot \mathbf{s}')$ for the given LPN sample $(\mathbf{A}, \mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e} \bmod 2)$. The last one is to replace the i-th column vector of $\mathbf{A}$ with a random vector. Suppose that the i-th entry of $\mathbf{s} + \mathbf{s}'$ is identical to the zero, it is still a LPN sample as well, if not it becomes a random pair. Hence, an oracle for decision LPN problems can recover the i-th entry by trying the above process for all underlying space. The last part makes this algorithm ineffective to the LPN over large fields. this search to decision algorithm should be repeated q-times. In particular, for $q \geq 4t$, [49] proposes that adapted SD 2.0 algorithm.

Lemma A.1 ([23, 49]) *The adapted SD 2.0 algorithm can solve the decisional (m, n, t)-LPN over $\mathbb{F}_q$ in time T with constant advantage, where $w = w(s) \in \mathbb{N}$ and $q \geq 4t$. The formula for T is given by*

$$T = \min_s \left(T_1 \cdot \left(\frac{\binom{m}{t}}{\binom{m-w}{t}} \cdot \frac{q}{q-1} \right)^2 + s \cdot \log q \cdot q^s \right).$$

Here, T_1 represents the time it takes to find one parity check vector. Following [14, 49], we also set $w = n - s + 1$ and $T_1 = n + 1$. As a result, we have

$$T = \min_s \left((n+1) \cdot \left(\frac{\binom{m}{t}}{\binom{m-n+s-1}{t}} \cdot \frac{q}{q-1} \right)^2 + s \cdot \log q \cdot q^s \right).$$

In summary, the prior approaches including BKW, ISD, and LPC algorithms heavily relies on the size of q and it is not competitive when q has an exponential size in the security parameter λ.