Report on Academic Activities

Sreyosi Bhattacharyya

Indian Statistical Institute, Kolkata

During the period of July 1, 2025 to the present day I have worked on the following areas:

- Information Set Decoding algorithm (ISD): Worked on a time/memory trade-off problem which has been described in Section 1. Previously, I had submitted a very initial version of this work. The current one gives more details. However, this work needs some more time to be fully completed.
- Explored an area outside my Ph.D,. thesis topic: provable security of symmetric key constructions. Specifically, I have studied research works in:
  - Proof techniques
    - H-coefficient technique: Studied this survey paper [11]
    - A new proof technique called power bounds [3] proposed in Crypto 2025 based on hypothesis testing.
  - Key Control Security [4]

The papers on power bound [3]and key control security [4, 5, 21] have been suggested to me by Prof. Mridul Nandi when I interacted with him after studying H-technique. At present I am looking into multi-user security some symmetric-key constructions.

1 Problem on ISD

Abstract. Information Set Decoding (ISD) algorithm proposed by May et al. (MMT) performs two-level collision search using *meet-in-the-middle* approach in both the levels. In Eurocrypt'23 Esser and Zweydinger proposed a time/memory trade-off of the MMT algorithm by replacing the *meet-in-the-middle* in the first level by Schroeppel-Shamir algorithm. The Schroeppel-Shamir algorithm takes same time as that of *meet-in-the-middle* to solve an instance of 0/1-knapsack problem but it does so with a reduced memory complexity than that of *meet-in-the-middle*. For the present work we perform a tighter analysis of the Esser-Zweydinger time/memory trade-off of MMT using the simpler heuristic version of Schroeppel-Shamir algorithm proposed by Howgrave-Graham and Joux (HGJ) in Eurocrypt'10. Our tighter analysis is due to the following observations: *1. The computation tree of* HGJ *version is a height-2 binary tree. 2. When* HGJ *is plugged into* MMT*, unlike the classical* 0/1*-knapsack problem, the second filtering length of* HGJ *is naturally a parameter of* ISD *which can be optimized for a given triplet: code-length, dimension of*

the code and weight of error-vector. 3. Replacing meet-in-the-middle by HGJ *leaves more scope of sub-tee repetition.* These observations in the context of ISD to helped us specifying ranges of possible values for the parameters of MMT algorithm. Along with these observations we have given detailed mathematical arguments to prove the exact number of representations and number of iterations which give the tighter analysis. As an application of our algorithm we study the time and memory complexity estimates of Classic McEliece for three different memory access costs. We have obtained reduced time complexity estimates for all the categories for every memory access models. The memory requirement of our algorithm for categories 1 never exceed 2^{60} bits for logarithmic and cube-root access model and that of Category 3 does not exceed 2^{60} for cube-root access model.
Keywords: Information Set Decoding, code-based cryptography, Classic McEliece, time/memory trade-off, representation technique, 0/1-knapsack problem, concrete analysis

2 Introduction

Information Set Decoding (ISD) is a generic decoding technique which forms the most useful cryptanalytic tool against code-based encryption schemes when the underlying linear code is treated as a random linear code. The use of *meet-in-the-middle* subroutines in ISD has been introduced by Stern in [22]. Subsequently more advanced ISD algorithms like [16, 2] have used *meet-in-the-middle* subroutines. Esser and Zweydinger in [9] have proposed a time/memory trade-off of the algorithm in [16]. This time/memory trade-off has been achieved by replacing the *meet-in-the-middle* subroutines of [16] by new subroutines obtained due to an algorithm proposed by Schroeppel and Shamir in [20] which solves an instance of 0/1-knapsack problem with same time complexity as the *meet-in-the-middle* but with reduced memory complexity. The Schroeppel-Shamir algorithm needs large priority queues. Howgrave-Graham and Joux have proposed a simpler heuristic alternative version of the Schroeppel-Shamir algorithm in [10] which does not need to handle large priority queues.

2.1 Motivation

The HGJ algorithm is heuristic in nature. The computation tree is a depth-2 binary tree. Additionally it does *bitwise collision checking*. The Howgrave-Graham-Joux alternative for [20] can be expressed as a depth-2 binary tree. A number σ is chosen randomly from the set $\{0, 1, \ldots, 2^{N/4} - 1\}$. There are four initial/baselists. For each $x \in L_1$ and $y \in L_2$ perform $x + y \bmod 2^{N/4}$ and keep those entries where $x + y \equiv \sigma \bmod 2^{N/4}$. Similarly for L_3 and L_4 check whether $x' + y' \equiv W - \sigma \bmod 2^{N/4}$ where W is the target. The next filtering step is naturally done on the rest of the bits.

The aspects of the HGJ version which is of our interest are:

– Introduction of a mod size of baselist modulus value for the first collision checking or filtering. Size of the baselist/initial lists for 0/1-knapsack problem is $2^{N/4}$.

– Introducing bitwise collision checking for Schroeppel-Shamir algorithm.

In the context of ISD the above points naturally imply the following points:

– Using a mod size of baselist target value in vector version essentially means that the target vector is of a length equal to $\log_2$(size of baselist). This keeps the expected size of a list obtained after the first filter equal to size of baselist.

– The bitwise collision checking of HGJ algorithm helps to further control the target vector's length in the context of MMT [16]. This is because when the HGJ algorithm is used in to replace the first *meet-in-the-middle* of MMT the number of bit positions to be checked in the second level of HGJ is too a parameter which can be chosen from a specific range of possible values for a given code.

Additionally our analysis is based on the following observations.

– Schroeppel-Shamir [20] or HGJ [10] requires the input set to be divided in to four equal parts.

– But unlike the classical 0/1-knapsack problem MMT requires to meet certain weight distribution constraint.

– The length of the target vector for the second level of HGJ in context of ISD is a parameter itself. Along with this observation plugging in HGJ in MMT gives more scope of sub-tree repetitions.

Our Contributions. We propose a tighter analysis of the Esser-Zweydinger time/memory trade-off of MMT. We have given the detailed mathematical arguments behind our analysis. To obtain these time and memory analyses, we state the *size of the initial lists/ baselists, the exact number of representations and expected number of iterations to get success* and *combinatorial arguments* have been given to prove the correctness of these expressions. As an application of our work we study the concrete analyses of time and memory bit complexities when these modifications are made in the Schroeppel-Shamir subroutine calls of [9]. We have observed improved time/memory trade-off of [16] for all the five categories and all types of memory access costs of *Classic McEliece*. The difference between time estimates in bits to break AES (143, 207 or 272) and our algorithm is more than the difference in Esser-Zweydinger [9]. The memory requirements of our algorithm for NIST category 1 never exceed 2^{60} bits for logarithmic and cube-root access model and that of Category 3 does not exceed 2^{60} for cube-root access model.

2.2 Related Works

The first ISD algorithm was proposed by Prange [19] in 1962. Subsequently improvements were proposed in [14, 15, 22, 8, 16, 2, 17, 6, 13]. Among these works the use of standard Meet-in-the-Middle (MITM) technique in ISD was introduced by Jacques Stern in [22]. The following advanced works in [16] and [2] use representation technique of [10] along with the MITM. In Eurocrypt 2023, Esser and Zweydinger [9] proposed a time/memory trade-off of [16]. This novel work replaced MITM collision finding routines with an algorithm proposed by Schroeppel and Shamir in [20] to solve the 0/1-knapsack problem.

The meet-in-the-middle algorithm finds a solution of the 0/1-knapsack problem for an input set of size N in $O(2^{N/2})$ time with $O(2^{N/2})$ memory while the algorithm proposed by Schroeppel and Shamir takes same time but a reduced memory $O(2^{N/4})$. However, the original Schroeppel and Shamir's algorithm requires large priority queues. Howgrave-Graham and Joux proposed an alternative version of the algorithm by Schroeppel and Shamir in [10] which does not need to use priority queues. This version does not provide any improvement theoretically in terms of time or memory complexities. They have proposed the new version which need not handle large priority queues.

Another work [12] gives time/memory trade-off of ISD algorithms but this work focuses on large-weight ternary error vectors. This work combines Wagner's generalized birthday problem [23] with representation technique.

3 Notations, Abbreviations and Preliminaries

Throughout the paper vectors are considered as row vectors and denoted by bold lower case letters and matrices are denoted by bold upper case letters. Identity matrices are denoted by $\mathbf{I_j}$ where j is the order of the matrix. $\mathbf{0}_j$ and $\mathbf{1}_j$ denote respectively an all-zero and all-one vector of length j. The cardinality of any finite set $\mathcal{S}$ will be denoted as $|\mathcal{S}|$. The distance metric considered is hamming metric and hamming weight of any vector $\mathbf{v}$ is denoted as $|\mathbf{v}|$. For any positive integer i, $[i]$ denotes the set $\{1, \ldots, i\}$. If two matrices $\mathbf{M}_1$ and $\mathbf{M}_2$ have equal number of rows then by $[\mathbf{M}_1|\mathbf{M}_2]$ we will denote the matrix obtained by juxtaposing $\mathbf{M}_1$ and $\mathbf{M}_2$. For any vector $\mathbf{v}$ of length j, $\mathbf{v}_{[i]}$ such that $i < j$ denotes the projection of the positions of $\mathbf{v}$ corresponding to $[i]$. Information Set Decoding, May-Meurer-Thomae algorithm [16], Schroeppel-Shamir and Howgrave-Graham-Joux have been abbreviated as ISD, MMT, SS and HGJ respectively. To refer to the work by Esser and Zweydinger in [9] we use the abbreviation EZ.

Let n and k be two positive integers such that $n > k$ and $\mathcal{C}$ be a subspace of dimension k of a vector space $\mathcal{V} = \{\mathbf{x} : \mathbf{x} \in \mathbb{F}_2^n\}$, where $\mathbb{F}_2$ is a finite field of two elements. Let the basis vectors of $\mathcal{C}$ be given by a matrix $\mathbf{G}_{k\times n}$. Let $\mathbf{H}_{(n-k)\times n}$ be another matrix such that $\mathbf{G} \cdot \mathbf{H}^\intercal = \mathbf{0}^\intercal$. Let d be the maximum hamming distance between any two vectors $(x), (y) \in \mathcal{C}$. $\mathcal{C}(n,\ k,\ d)$ is a linear code over $\mathbb{F}_2$

of length n, dimension k and minimum distance d with $\mathbf{G}_{k\times n}$ as the generator matrix and $\mathbf{H}_{(n-k)\times}n$ as the parity-check matrix.

Definition 1 Computational Syndrome Decoding Problem (CSDP) Let $n,\ k$ and w be three positive integers such that $n > k$. Let $\mathbf{H}_{(\mathbf{n-k})\times\mathbf{n}}$ be a matrix such that $\forall i \in \{1,\ldots,n-k\}$ and $j \in \{1,\ldots,n\}$ $H_{ij} \in \mathbb{F}_2$ and $\mathbf{s}^\intercal \in \mathbb{F}_2^{n-k}$. The goal is to find $\mathbf{e}^\intercal \in \mathbb{F}_2^n$ such that $\mathbf{H}\cdot\mathbf{e}^\intercal = \mathbf{s}^\intercal$ and $|\mathbf{e}| = w$.

So, the triplet $(\mathbf{H}, \mathbf{s}, \omega)$ forms an instance of Computational Syndrome Decoding problem.

Information Set Decoding. The aim of an ISD algorithm is to solve an instance of CSDP. The syndrome vector is the sum of the ω columns of $\mathbf{H}$ corresponding to the *one* positions of $\mathbf{e}$.

However, ISD reduces the dimension of the problem to obtain a smaller instance. Using Gaussian elimination on $\mathbf{H}_{(n-k)\times n}$ the following form is obtained-

$$\begin{bmatrix} \mathbf{A}_1 & \mathbf{A}_2 & \mathbf{0}_{\ell\times(r-\ell)} \\ \mathbf{B}_1 & \mathbf{B}_2 & \mathbf{I}_{(r-\ell)} \end{bmatrix}$$

$$\mathbf{H}' = \begin{bmatrix} \mathbf{A}_1 & \mathbf{A}_2 \\ \mathbf{B}_1 & \mathbf{B}_2 \end{bmatrix}$$

There two equations now- $\mathbf{A}\cdot\mathbf{e}_1^\intercal = \mathbf{s}_1^\intercal$ and $\mathbf{B}\cdot\mathbf{e}_1^\intercal + \mathbf{e}_2^\intercal = \mathbf{s}_2^\intercal$ where $\mathbf{A} = [\mathbf{A}_1|\mathbf{A}_2]$ and $\mathbf{B} = [\mathbf{B}_1|\mathbf{B}_2]$. The first equation is solved to obtain $\mathbf{e}_1$. MMT solves the first equation using *meet-in-the-middle* and *representation technique* [10]. The problem of obtaining $\mathbf{e}_1$ has been named as *submatrix matching problem* by May et al. in [16].

MMT[16] We give a brief description of MMT [16]. MMT uses *representation technique* introduced in [10]. MMT divides the information set into two halves. Here the size of information set is $(k+\ell)$. MMT requires four baselists. Each baselist has one entry corresponding to each of the $\binom{(k+\ell)/2}{p/4}$ combinations. Let us denote the i-th baselist as L_{1i} where $i \in \{1,\ 2,\ 3,\ 4\}$ and let $\mathcal{I}$ be the information set and $|\mathcal{I}| = p$. Let $\mathcal{I} = \mathcal{I}_1 \cup \mathcal{I}_2$ and $|\mathcal{I}_1| = |\mathcal{I}_2| = p/2$. For $j \in 1, 3$, $L_{1j} = \{I_{1j}, \sum_{i\in I}\mathbf{H}'_i,$ where $I_{1j} \subset \{1,\ldots,(k+\ell)/2\}$ and $|I_{1j}| = p/4\}$ and for $j \in 2, 4$, $L_{1j} = \{I_{1j}, \sum_{i\in I}\mathbf{H}'_i,$ where $I_{1j} \subset \{(k+\ell)/2+1,\ldots,(k+\ell)\}$ and $|I_{1j}| = p/4\}$. Following the meet-in-the-middle technique, the second components of L_{11} and L_{12} are added mod 2 and then filtered based on ℓ_1 positions. The new tuples are stored in L_1. Similarly we obtain L_2 from L_{13} and L_{14}. The expected number of tuples in L_1 is $\max\{L_{11}, (|L_{11}|\cdot|L_{12}|)/2^{\ell_1}\}$ and in L_2 is $\max\{L_{13}, (|L_{13}|\cdot|L_{14}|)/2^{\ell_1}\}$. For each of L and R MMT chooses $p/4$ one-positions from one half and the another $p/4$ one-positions from the other half of the information set. Thus finally one solution has $\binom{p/2}{p/4}^2$ representations. The partial error vectors corresponding to these second level lists forms a subset of $(k+\ell)/2$ length vectors with weight $p/2$. The last merge of L_1 and L_2 gives us partial error vectors with $k+\ell$ length and p weight.

Probability of success per iteration of MMT= $\frac{\binom{(k+\ell)/2}{p/2}^2 \cdot \binom{n-k-\ell}{w-p}}{\binom{n}{w}}$

Stirling's Approximation. The term $\binom{a}{b}$ can be approximated using Stirling's approximation as $2^{a \cdot H(\frac{b}{a})}$ where $H(\frac{b}{a}) = -\frac{b}{a} \cdot \log_2(\frac{b}{a}) - (1 - \frac{b}{a}) \cdot \log_2(1 - \frac{b}{a})$ is the binary entropy function.

4 New Conditions On the Filtering Lengths for a better Adoption HGJ in MMT And Description of the New Algorithm

4.1 Constraints

In ISD the match-then-filter is done on ℓ number of bits where ℓ is a parameter of the corresponding ISD algorithm. The SS is not applied on the entire ℓ bits in [9]. The work in [9] breaks ℓ as $\ell = \ell' + \ell''$ or in other words ℓ is decomposed into two parts where ℓ' forms the matching length of the Schroeppel-Shamir subroutines. We decompose ℓ into three parts and write ℓ as follows.

$$\ell = \ell_1 + \ell_2 + \ell_3 \tag{1}$$

where ℓ_1 and ℓ_2 are number of bits to be matched respectively in the first and second level of the HGJ tree. The value of ℓ_1 is $\log_2 |L_{base}|$. Let each of the four lists be of size 2^{ℓ_1} and each sum vector of these lists are of size ℓ bits. In the first level the match-and-filter is done on ℓ_1 bits and the next filter is on ℓ_2 bits.

$$\frac{2^{2 \cdot \ell_1}}{2^{\ell_2}} \geq 1$$
$$or,\ 2^{2 \cdot \ell_1 - \ell_2} \geq 2^0$$

Taking $\log_2$ on both sides,

$$2 \cdot \ell_1 - \ell_2 \geq 0$$
$$or,\ \ell_1 \geq \frac{\ell_2}{2}$$
$$or,\ \ell_2 \leq 2 \cdot \ell_1 \tag{2}$$

The target length for the last filter is ℓ_3 where $\ell_3 = \ell - \ell_1 - \ell_2$ such that $\ell_3 > 0$. The last constraint to be fulfilled is:

$$\frac{2^{4 \cdot \ell_1}}{2^{2 \cdot \ell_2 + \ell_3}} \geq 1. \tag{3}$$

Taking $\log_2$ on both sides of the above equation we get the following.

$$4 \cdot \ell_1 - 2 \cdot \ell_2 - \ell_3 \geq 0 \tag{4}$$

Let $|L_{base}|$ be ξ. The first level of HGJ-version of SS must match on $\log_2 \xi$ bits.

$$\ell_1 = \log_2 \xi \tag{5}$$

By 2, if the number of bits ℓ_2 to be matched on the next level is greater than $2 \cdot \ell_1$ then the whole process must be repeated ξ times. If the number of bits ℓ_2 to be matched on the next level is lesser than $2 \cdot \ell_1$ then the expected size of collision list is at least one.

4.2 Using The Modified HGJ algorithm To Get A New TMTO

Schroeppel-Shamir divides the input set of size N into four parts of size $N/4$ each. Schroeppel-Shamir technique does not have any restriction on weight of the output. In case of MMT there is a restriction on weight. The weight of the output vector must be $p/2$ for each call to the Schroeppel-Shamir sub-routine as in [9].

MMT requires baselists of size $\binom{(k+\ell)/2}{p/4} \approx 2^{\frac{(k+\ell)}{2} \cdot H(\frac{p}{2 \cdot (k+\ell)})}$. These lists forms the inputs for *meet-in-the-middle* subroutines. So the new baselist size for our algorithm should be $2^{\frac{(k+\ell)}{4} \cdot H(\frac{p}{2 \cdot (k+\ell)})}$.

We divide the information set into four parts of size $(k+\ell)/4$ each. Four knapsacks are formed for constructing each of L and R. Thus we have eight baselists. We must equally divide the one-positions among these parts. Otherwise we end up with unbalanced knapsacks. Detailed discussions on unbalanced knapsacks are there in [1, 10]. For constructing the knapsacks we consider only those combinations where $p/8$ ones are present instead of listing all the $2^{(k+\ell)/4}$ possible combinations. The size of each knapsack or baselist is thus $\binom{(k+\ell)/4}{p/8} \approx 2^{\frac{(k+\ell)}{4} \cdot H(\frac{p}{2 \cdot (k+\ell)})}$. So,

$$\xi = \binom{(k+\ell)/4}{p/8}.$$

Algorithm 1 and Algorithm 2 describe the basic steps to obtain L and R for MMT using the modified HGJ. The $\mathsf{LinAlg}(\cdot)$ function is responsible for obtaining the semi-systematic form of the parity-check matrix. For a subset $\mathcal{S}$ of $[k+\ell]$, by $\chi(S)$ we will denote the k-bit string $\mathbf{x} = (x_1, \ldots, x_k)$ such that $x_i = 1$ if and only if $i \in \mathcal{S}$.

The baselists are named as L_{11}, L_{12}, L_{13}, L_{14}, R_{11}, R_{12}, R_{13} and R_{14}. The lists L_{11}, L_{13}, R_{11}, R_{13}, L_1, R_1 and L are stored. Rest of the lists are read on-the fly. For $i \in \{1, 2, 3, 4\}$ L_{1i} is enumerated based on $\mathcal{I}_{1i}$ and R_{1i} is enumerated based on $\mathcal{I}_{2i} \subset [(i-1)(k+\ell)/4 + 1, i(k+\ell)/4] - \mathcal{I}_{1i}$ where $|\mathcal{I}_{2i}| = p/8$.

$$\begin{bmatrix} \mathbf{A}_1 & \mathbf{A}_2 & \mathbf{A}_3 & \mathbf{A}_4 & \mathbf{0}_{\ell\times(r-\ell)} \\ \mathbf{B}_1 & \mathbf{B}_2 & \mathbf{B}_3 & \mathbf{B}_4 & \mathbf{I}_{(r-\ell)} \end{bmatrix} \begin{bmatrix} \mathbf{e}_1^\top \\ \mathbf{e}_2^\top \\ \mathbf{e}_3^\top \\ \mathbf{e}_4^\top \\ \mathbf{e}_5^\top \end{bmatrix}.$$

Fig. 1: Representation of the parity-check matrix. In this figure $\mathbf{A}_1,\ \mathbf{A}_2,\ \mathbf{A}_3$ and $\mathbf{A}_4$ are $\ell \times (k+\ell)/4$ matrices and $\mathbf{B}_1,\ \mathbf{B}_2,\ \mathbf{B}_3$ and $\mathbf{B}_4$ are $(n-k-\ell) \times (k+\ell)/4$ matrices.

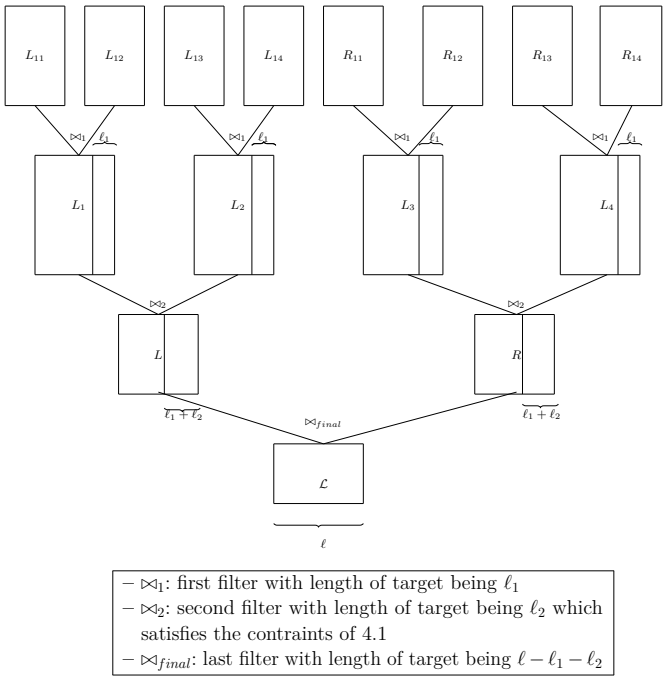

Fig. 2: Construction of the lists L and R for MMT using the new subroutine for Schroeppel-Shamir.

Please refer to Figure 1 and Figure 2 for the ease of following the description.

$$L_{11} = \{(\mathcal{I}_{11}, \mathbf{a}_{11}, \mathbf{b}_{11}) : |\mathcal{I}_{11}| = p/8,\ \mathcal{I}_{11} \subset [1, (k+\ell)/4],\ \mathbf{a}_{11} = \sum_{i \in \mathcal{I}_{11}} \mathbf{A}_{1i}$$

$$\mathbf{b}_{11} = \sum_{i \in \mathcal{I}_{11}} \mathbf{B}_{1i}\}$$

$$R_{11} = \{(\mathcal{I}_{21}, \mathbf{a}_{21}, \mathbf{b}_{21}) : |\mathcal{I}_{21}| = p/8,\ \mathcal{I}_{21} \subset [1, (k+\ell)/4],\ \mathbf{a}_{21} = \sum_{i \in \mathcal{I}_{21}} \mathbf{A}_{1i},$$

$$\mathbf{b}_{21} = \sum_{i \in \mathcal{I}_{21}} \mathbf{B}_{1i}\}$$

$$L_{12} = \{(\mathcal{I}_{12}, \mathbf{a}_{12}, \mathbf{b}_{12}) : |\mathcal{I}_{12}| = p/8,\ \mathcal{I}_{12} \subset [(k+\ell)/4+1, (k+\ell)/2],$$

$$\mathbf{a}_{12} = \sum_{i \in \mathcal{I}_{12}} \mathbf{A}_{2i},\ \mathbf{b}_{12} = \sum_{i \in \mathcal{I}_{12}} \mathbf{B}_{2i}\}$$

$$R_{12} = \{(\mathcal{I}_{22}, \mathbf{a}_{22}, \mathbf{b}_{22}) : |\mathcal{I}_{22}| = p/8\ \mathcal{I}_{22} \subset [(k+\ell)/4+1, (k+\ell)/2],$$

$$\mathbf{a}_{22} = \sum_{i \in \mathcal{I}_{22}} \mathbf{A}_{2i}, \mathbf{b}_{22} = \sum_{i \in \mathcal{I}_{22}} \mathbf{B}_{2i}\}$$

$$L_{13} = \{(\mathcal{I}_{13}, \mathbf{a}_{13}, \mathbf{b}_{13}) : |\mathcal{I}_{13}| = p/8,\ \mathcal{I}_{13} \subset [(k+\ell)/2+1, 3\cdot(k+\ell)/4],$$

$$\mathbf{a}_{13} = \sum_{i \in \mathcal{I}_{13}} \mathbf{A}_{3i},\ \mathbf{b}_{13} = \sum_{i \in \mathcal{I}_{13}} \mathbf{B}_{3i}\}$$

$$R_{13} = \{(\mathcal{I}_{23}, \mathbf{a}_{23}, \mathbf{b}_{23}) : |\mathcal{I}_{23}| = p/8,\ \mathcal{I}_{23} \subset [(k+\ell)/2+1, 3\cdot(k+\ell)/4],$$

$$\mathbf{a}_{23} = \sum_{i \in \mathcal{I}_{23}} \mathbf{A}_{3i},\ \mathbf{b}_{23} = \sum_{i \in \mathcal{I}_{23}} \mathbf{B}_{3i}\}$$

$$L_{14} = \{(\mathcal{I}_{14},\ \mathbf{a}_{14},\ \mathbf{b}_{14}) : |\mathcal{I}_{14}| = p/8,\ \mathcal{I}_{14} \subset [3\cdot(k+\ell)/4+1, (k+\ell)],$$

$$\mathbf{a}_{14} = \sum_{i \in \mathcal{I}_{14}} \mathbf{A}_{4i},\ \mathbf{b}_{14} = \sum_{i \in \mathcal{I}_{24}} \mathbf{B}_{4i}\}$$

$$R_{14} = \{(\mathcal{I}_{24},\ \mathbf{a}_{24},\ \mathbf{b}_{24}) : |\mathcal{I}_{24}| = p/8,\ \mathcal{I}_{24} \subset [3\cdot(k+\ell)/4+1, (k+\ell)],$$

$$\mathbf{a}_{24} = \sum_{i \in \mathcal{I}_{24}} \mathbf{A}_{4i},\ \mathbf{b}_{24} = \sum_{i \in \mathcal{I}_{24}} \mathbf{B}_{4i}\}$$

Collisions obtained in the first level are stored in four lists L_1, L_2, R_1 and R_2.

$$L_1 = \{(\mathcal{I}_1, \mathbf{a}_1, \mathbf{b}_1) : \mathcal{I}_1 = \mathcal{I}_{11} \cup \mathcal{I}_{12},\ \mathbf{a}_1 = \mathbf{a}_{11} \oplus \mathbf{a}_{12},\ \mathbf{a}_{11} \oplus \mathbf{a}_{12} = \mathbf{t}_1 \in F_2^{\ell_1}\}$$
$$L_2 = \{(\mathcal{I}_2, \mathbf{a}_2, \mathbf{b}_2) : \mathcal{I}_2 = \mathcal{I}_{13} \cup \mathcal{I}_{14},\ \mathbf{a}_2 = \mathbf{a}_{13} \oplus \mathbf{a}_{14},\ \mathbf{a}_{13} \oplus \mathbf{a}_{14} = 0 - \mathbf{t}_1\}$$
$$R_1 = \{(\mathcal{I}_3, \mathbf{a}_3, \mathbf{b}_3) : \mathcal{I}_3 = \mathcal{I}_{21} \cup \mathcal{I}_{22},\ \mathbf{a}_3 = \mathbf{a}_{21} \oplus \mathbf{a}_{22},\ \mathbf{a}_{21} \oplus \mathbf{a}_{22} = \mathbf{t}_2,\ \mathbf{t}_2 \in \mathbb{F}_2^{\ell_1}\}$$
$$R_2 = \{(\mathcal{I}_4, \mathbf{a}_4, \mathbf{b}_4) : \mathcal{I}_4 = \mathcal{I}_{23} \cup \mathcal{I}_{24},\ \mathbf{a}_4 = \mathbf{a}_{23} \oplus \mathbf{a}_{24},\ \mathbf{a}_{23} \oplus \mathbf{a}_{24} = \mathbf{s}_{[\ell_1]} - \mathbf{t}_2\}$$
$$L = \{(\mathcal{I}_L, \mathbf{a}_L, \mathbf{b}_L) : \mathcal{I}_L = \mathcal{I}_1 \cup \mathcal{I}_2, \mathbf{a}_L = \mathbf{a}_1 \oplus \mathbf{a}_2 = \mathbf{t}_{[\ell_1+\ell_2]},\ \mathbf{t} \in \mathbb{F}_2^{\ell_2},\ \mathbf{b}_L = \mathbf{b}_1 \oplus \mathbf{b}_2\}$$
$$R = \{(\mathcal{I}_R, \mathbf{a}_R, \mathbf{b}_R) : \mathcal{I}_R = \mathcal{I}_3 \cup \mathcal{I}_4, \mathbf{a}_R = \mathbf{a}_3, \mathbf{a}_3 \oplus \mathbf{a}_4 = \mathbf{s}_{[\ell_1+\ell_2]} + \oplus\mathbf{t},\ \mathbf{b}_R = \mathbf{b}_3 \oplus \mathbf{b}_4\}$$

The final list $\mathcal{L}$ is defined as-

$$\mathcal{L} = \{(\mathcal{I},\ \sum_{i\in\mathcal{I}} H_i) : \mathcal{I} = \mathcal{I}_L \cup \mathcal{I}_R,\ \sum_{i\in\mathcal{I}_L} H_i = \sum_{i\in\mathcal{I}_R} H_i + \mathbf{s}_{[\ell]}\}$$

Let rep be the number of representations of the solution.

- If $\ell_{filter} < rep$ then it implies that number of representations of the solution is greater than the total number of target vectors of length ℓ_{filter}. For each of the $2^{\ell_{filter}}$ random target values at least one representation survives through the filter according to the *Pigeonhole Principle*. So for this case at least one representation survives through the filter with probability *one*.
- If $\ell_{filter} > rep$ then the number of representations of the solution is lesser than the number of random target vectors. So a representation survives by probability $2^{rep}/2^{\ell_{filter}}$. Expected number of times the filter should be repeated independently so that a representation survives is $2^{\ell_{filter}}/2^{rep}$.

For the new algorithm-

- the first filter should be repeated $2^{\ell_1 - rep}$ number of times when $\ell_1 > rep$. The expected number of representations surviving through this filter is $2^{rep-\ell_1}$
- the second filter is repeated $2^{\ell_1+\ell_2-rep}$ number of times when $\ell_2 > rep-\ell_1$ because the expected number of representations arriving at the second filter is $2^{rep-\ell_1}$. The probability that at least one representation survives through the second filter is $2^{rep-\ell_1}/2^{\ell_2}$.

Algorithm 1 A general formulation of ISD algorithm using MITM.

Require: $(\mathbf{H}, \mathbf{s}, \omega)$
Ensure: $\mathbf{e}$ such that $\mathbf{He}^\top = \mathbf{s}^\top$ and $|\mathbf{e}| = \omega$.

while true **do**
  Choose a random permutation matrix $\mathbf{P}$ of order $n \times n$
  $(\mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3, \mathbf{A}_4, \mathbf{B}_1, \mathbf{B}_2, \mathbf{B}_3, \mathbf{B}_4, \mathbf{s}_1, \mathbf{s}_2) \leftarrow \mathsf{LinAlg}(\mathbf{H}, \mathbf{P}, \mathbf{s}, \ell)$
  $\mathcal{L} \leftarrow \mathsf{MMT} - \mathsf{modified} - \mathsf{HGJ}(\mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3, \mathbf{A}_4, \mathbf{B}_1, \mathbf{B}_2, \mathbf{B}_3, \mathbf{B}_4, \mathbf{s_1}, \mathbf{s_2}, \omega, p)$
  for all $(\mathcal{I}, \sum_{i\in\mathcal{I}} \mathbf{H}_i) \in \mathcal{L}$ **do**
    if $\mathbf{b}_L \oplus \mathbf{b}_R \oplus \mathbf{s}_2 = \omega - p$ **then**
      $(\mathbf{e}_1,\ \mathbf{e}_2,\ \mathbf{e}_3,\ \mathbf{e}_4) = \chi(\mathcal{I})$, $\mathbf{e}_5 = \mathbf{b}_L \oplus \mathbf{b}_R \oplus \mathbf{s}_2$, $\mathbf{e} = (\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \mathbf{e}_4, \mathbf{e}_5)$
      return $\mathbf{Pe}^\top$
    end if
  end for
end while

Algorithm 2 MMT-modified-HGJ

Require: Parity-check matrix $\mathbf{H}'$, $\mathbf{s}_1$, ω
Ensure: A non-empty list of partial error vector(s) $\mathcal{L}$

Choose optimal $\ell,\ \ell_1,\ \ell_2,\ p$
$rep = \log_2 \binom{p/4}{p/8}^4$
Enumerate four lists L_{11}, L_{13}, R_{11} and R_{13}.
repeat
    Choose a random vector $\mathbf{t}_1$ of length $\log_2(\binom{(k+\ell)/4}{p/8})$. Let $\ell_1 = \log_2(\binom{(k+\ell)/4}{p/8})$.
    For each $(\mathcal{I}_{12}, \mathbf{a}_{12}, \mathbf{b}_{12}) \in L_{12}$ generated on-the-fly check for $(\mathcal{I}_{11}, \mathbf{a}_{11}, \mathbf{b}_{11}) \in L_{11}$ such that $(\mathbf{a}_{11} \oplus \mathbf{a}_{12})_{[\ell_1]} = \mathbf{t}_1$
    if $(\mathbf{a}_{11} \oplus \mathbf{a}_{12})_{[\ell_1]} = \mathbf{t}_1$ **then**
        Store the combination in L_1
    end if
    For each $(\mathcal{I}_{14}, \mathbf{a}_{14}, \mathbf{b}_{14}) \in L_{14}$ generated on-the-fly check for $(\mathcal{I}_{13}, \mathbf{a}_{13}, \mathbf{b}_{13}) \in L_{13}$ such that $(\mathbf{a}_{13} \oplus \mathbf{a}_{14})_{[\ell_1]} = 0 - \mathbf{t}_1$
    if $(\mathbf{a}_{13} \oplus \mathbf{a}_{14})_{[\ell_1]} = 0 - \mathbf{t}_1$ **then**
        Store the combination in L_2
    end if
    repeat
        For each $(\mathcal{I}_{22}, \mathbf{a}_{22}, \mathbf{b}_{22}) \in R_{12}$ generated on-the-fly check for $(\mathcal{I}_{21}, \mathbf{a}_{21}, \mathbf{b}_{21}) \in R_{11}$ such that $(\mathbf{a}_{21} \oplus \mathbf{a}_{22})_{[\ell_1]} = \mathbf{t}_2$
        if $(\mathbf{a}_{21} \oplus \mathbf{a}_{22})_{[\ell_1]} = \mathbf{t}_2$ **then**
            Store the combination in R_1
        end if
        For each $(\mathcal{I}_{24}, \mathbf{a}_{24}, \mathbf{b}_{24}) \in R_{14}$ generated on-the-fly check for $(\mathcal{I}_{23}, \mathbf{a}_{23}, \mathbf{b}_{23}) \in R_{13}$ such that $(\mathbf{a}_{23} \oplus \mathbf{a}_{24})_{[\ell_1]} = \mathbf{s}_{[\ell_1]} \oplus \mathbf{t}_2$
        if $(\mathbf{a}_{23} \oplus \mathbf{a}_{24})_{[\ell_1]} = \mathbf{s}_{[\ell_1]} \oplus \mathbf{t}_2$ **then**
            Store the combination in R_2
        end if
        repeat
            For each $(\mathcal{I}_1, \mathbf{a}_1, \mathbf{b}_1) \in L_1$ check for $(\mathcal{I}_2, \mathbf{a}_1, \mathbf{b}_1) \in L_2$ such that $(\mathbf{a}_1 \oplus \mathbf{a}_2)_{[\ell_1+\ell_2]} = \mathbf{t}$
            if $(\mathbf{a}_1 \oplus \mathbf{a}_2)_{[\ell_1+\ell_2]} = \mathbf{t}$ **then**
                Store the combination in L
            end if
            For each $(\mathcal{I}_3, \mathbf{a}_3, \mathbf{b}_3) \in L_3$ check for $(\mathcal{I}_4, \mathbf{a}_4, \mathbf{b}_4) \in L_4$ such that $(\mathbf{a}_3 \oplus \mathbf{a}_4)_{[\ell_1+\ell_2]} = \mathbf{t}$
            if $(\mathbf{a}_3 \oplus \mathbf{a}_4)_{[\ell_1+\ell_2]} = \mathbf{t}$ **then**
                Store the combination in R
            end if
            Check for collision between L and R on the remaining ℓ_3 bits.
            if $(\mathbf{a}_L \oplus \mathbf{a}_R)_{[\ell_1+\ell_2+\ell_3]} = \mathbf{s}_{[\ell_1+\ell_2+\ell_3]}$ **then**
                Store the combination in $\mathcal{L}$
            end if
            if $\mathcal{L}$ is not empty **then**
                return $\mathcal{L}$
            end if
        until $2^{\max\{\ell_1+\ell_2-rep,0\}}$ ▷ Repeated when solution is not obtained
    until $2^{\max\{\ell_1-rep,0\}}$ ▷ Repeated when the first SS is repeated
until $2^{\max\{\ell_1-rep,0\}}$ ▷ Repeated when solution is not obtained repeating the lower subtrees

5 Time and Memory Complexities

In this section we discuss the time and memory complexities of the new algorithm. In [9] time and memory complexities are in terms of bits. If a is the total number of field operations then the bit complexity estimate of time is $\log_2(a) + \log_2(n)$. The estimates obtained using this method can be found in Table 2. Before beginning with this discussion we briefly describe the complexity estimates of [9] for the sake of completeness.

Time and memory bit complexity estimates of [9] The vectors in the baselists of MMT[16] belongs to the set $\mathcal{D}_0 = \{e'' : e'' \in \mathbb{F}^{(k+l)/2},\ |e''| = p/4\}$. After the meet-in-the-middle collision the sum vectors belong to the set $\mathcal{D}_1 = \{e' : e' \in \mathbb{F}^{(k+l)},\ |e'| = p/2\}$. The work in [9] expresses the time and memory estimates exactly as-

$$T_{\mathsf{EZ}} = 2^{\ell' - rep} \cdot \max\{|\mathcal{D}_1|^{1/2}, \frac{|\mathcal{D}_1|}{2^{\ell'}}, \frac{|\mathcal{D}_1|^2}{2^{\ell+\ell'}}\} \tag{6}$$

$$M_{\mathsf{EZ}} = \max\{|\mathcal{D}_1|^{1/4}, \frac{|\mathcal{D}_1|}{2^{\ell'}}\} \tag{7}$$

5.1 Time and memory bit complexity estimates of the new algorithm.

Now we give the expressions of time and memory complexity estimates of the new algorithm. Let the first level of filtering takes time HGJ_{11} and HGJ_{12} respectively for the two calls.

$$T_{\mathsf{HGJ}_{11}} = T_{\mathsf{HGJ}_{12}} = 2 \cdot |\mathcal{D}_1|^{1/4} + \frac{|\mathcal{D}_1|^{1/2}}{2^{\ell_1}} \tag{8}$$

The expected size of the list after the first filter is

$$|\mathcal{D}_1|^{1/4}, \tag{9}$$

Let the second level of filtering takes time HGJ_1 and HGJ_2.

$$T_{\mathsf{HGJ}_1} = T_{\mathsf{HGJ}_2} = 2 \cdot |\mathcal{D}_1|^{1/4} + \frac{|\mathcal{D}_1|^{1/2}}{2^{\ell_2}} \tag{10}$$

The expected size of a list after the second filter (in other words the expected list size after the calls to new Schroeppel-Shamir subroutines are complete) is-

$$M_{new-HGJ} = \max\{\mathcal{D}_1^{1/4}, \frac{\mathcal{D}_1{}^{1/2}}{2^{\ell_2}}\} \tag{11}$$

The time taken for the final merging is

$$T_{final} = 2 \cdot M_{new-HGJ} + \frac{M_{new-HGJ}{}^2}{2^{\ell_3}}. \tag{12}$$

The computation tree of the new algorithm is of depth-3. So we compute the total time taken by the new algorithm is obtained using the subtree repetition of [9]. If a representation is not obtained at the last level then the second level of the two HGJ's are repeated $\max\{0, \ell_1 + \ell_2 - rep\}$ times. If a solution is still not obtained we repeat the higher levels similarly.

Let $\max\{0, \ell_1 - rep\}$ be denoted by m_1 and $\max\{0, \ell_1 + \ell_2 - rep\}$ be denoted by m_2.

$$T_{total} = 2^{m_1} \cdot (2 \cdot T_{\mathsf{HGJ}_{11}} + 2^{m_1} \cdot (2 \cdot T_{\mathsf{HGJ}_{21}} + (2^{m_2} \cdot \underbrace{(T_{\mathsf{HGJ}_1} + T_{\mathsf{HGJ}_2} + T_{final})}))) \tag{13}$$

We compute the time complexity estimates using equation 13.

Taking the dominating terms of equation the time estimate may be written as-

$$T_{new} = 2^{3 \cdot \ell_1 + \ell_2 - 3 \cdot r} \cdot \max\{|\mathcal{D}_1|^{1/4}, \frac{|\mathcal{D}_1|^{1/2}}{2^{\ell_2}}, \frac{|\mathcal{D}_1|}{2^{\ell + \ell_1 - \ell_2}}\} \tag{14}$$

$$M_{new-SS} = \max\{|\mathcal{D}_1|^{1/4}, \frac{|\mathcal{D}_1|^{1/2}}{2^{\ell_2}}\} \tag{15}$$

We state a simple combinatorial identity before proceeding further with the analysis.

Lemma 1. *For any positive integers $a,\ b,\ c$*

$$\binom{a}{b+c} = \binom{a}{b} \cdot \binom{a-b}{c} / \binom{b+c}{c}$$

Lemma 2. *For the respective ranges of the parameters p and ℓ the probability of success per iteration is given by $\frac{\binom{(k+\ell)/4}{p/4}^4 \cdot \binom{n-k-\ell}{w-p}}{\binom{n}{w}}$.*

Proof. The information set is chosen uniformly and independently in each iteration. So we can say that $\mathbf{P}$ is also chosen uniformly and independently in each iteration and applied on the columns of the parity-check matrix $\mathbf{H}$. The total number of permutation is $n!$. Let $\mathcal{P}$ be the set of permutations which when applied on the columns of $\mathbf{H}$ give success. Thus the success probability in a single iteration is given by

$$\frac{|\mathcal{P}|}{n!}.$$

This set can be constructed in the following manner. There are four $(k+\ell)/4$-length parts of the information set. In each of the four divisions of the information

set distribute $p/8$ ones to a subset of the cells $1,\ 2,\ \ldots,\ (k+\ell)/4$. Distribute the remaining $p/8$ one positions in the remaining $(k+\ell)/4 - p/8$ cells. By Lemma 1 position of the ones can be fixed in $(\binom{(k+\ell)/4}{p/8}^4 \cdot \binom{(k+\ell)/4-p/8}{p/8}^4)/\binom{p/4}{p/8}^4$ ways. The remaining $n-\omega$ cells are filled wth zeroes. Now we have fixed positions of ones and zeroes in the n positions. The total number of such fixings is

$$\binom{(k+\ell)/4}{p/4}^4 \cdot \binom{n-k-\ell}{\omega-p}$$

The cells with zeroes and ones can be permuted among themselves in $\omega! \cdot (n-\omega)!$ ways. The size of $\mathcal{P}$ is $\binom{(k+\ell)/4}{p/4}^4 \cdot \omega! \cdot (n-\omega)! \cdot \binom{n-k-\ell}{\omega-p}$. So,

$$\pi = \frac{\binom{(k+\ell)/4}{p/4}^4 \cdot \binom{n-k-\ell}{\omega-p}}{\binom{n}{\omega}}$$

□

Lemma 3. *The exact number of representations is* $\binom{p/4}{p/8}^4$.

5.2 Numerical Results

The NIST call for proposals [18] for post-quantum cryptosystems outlines five categories. Of these, Categories 1, 3 and 5 associates to the security of AES-128, AES-192 and AES-256 and require cryptosystems to be secure under attacks using 2^{143}, 2^{207} and 2^{272} classical gates respectively. The description of five instances of Classic McEliece corresponding to the five categories are presented in Table 1.

We have considered three memory access models- constant, logarithmic and cube-root. Additionally for each of the models we have considered unbounded memory and memory settings upper bounded by 2^{60} and 2^{80} bits. Table 2 shows the new time complexities in terms of bits.

We obtain reduced time complexity estimates for all the categories for every memory access models. The memory requirement of our algorithm for categories 1 never exceed 2^{60} bits for logarithmic and cube-root access model and that of Category 3 does not exceed 2^{60} for cube-root access model.

The values in the columns marked by EZ have been obtained from Table 4 of [9].

category	n	k	w	$\log_2 P$	$\log_2 M_{\mathrm{mat}}$
1	3488	2720	64	20.99	21.35
3	4608	3360	96	22.00	22.46
5	6688	5024	128	22.00	23.41
	6960	5413	119	22.00	23.36
	8192	6528	128	23.37	23.70

Table 1: Parameters for Classic McEliece and the corresponding values of $\log_2 P$ and $\log_2 M_{\mathrm{mat}}$.

Table 2: Time-Memory Trade-off Points with constant memory access cost

memory bound	Category 1		Category 3		Category 5a		Category 5b		Category 5c	
	EZ [9]	new	EZ [9]	new	EZ [9]	new	EZ [9]	new	EZ [9]	new
no bound	142.02	140.70	181.91	176.80	248.18	240.30	247.49	239.12	277.37	270.04
≤ 60	145.47	140.70	188.16	176.84	263.16	248.74	283.64	250.84	298.65	284.34
≤ 80	143.14	140.70	184.06	176.80	258.58	240.30	258.87	239.15	293.4	270.14
logarithmic access model										
no bound	148.44	146.16	188.61	182.73	255.38	246.48	254.72	245.46	284.83	276.41
≤ 60	151.06	146.16	193.59	182.73	268.66	254.55	269.17	256.67	304.19	290.19
≤ 80	149.14	146.16	190.12	182.73	264.58	246.48	264.87	245.46	299.45	276.46
cube-root access model										
no bound	157.25	154.63	199.69	196.13	276.13	263.45	276.97	265.07	313.12	295.56
≤ 60	157.25	154.63	199.69	196.13	276.13	266.61	276.97	268.87	313.12	302.80
≤ 80	157.25	154.63	199.69	196.13	276.13	263.45	276.97	265.07	313.12	295.56

6 Conclusion and Future Work

We have proposed a modification of the Howgrave-Graham-Joux version of Schroeppel-Shamir algorithm [20]. This modification is relevant in the context of *Information Set Decoding* only. This work has given a new and improved time-memory trade-off for MMT algorithm [16] for all the five parameter sets of Classic McEliece. A comprehensive asymptotic analysis following the lines of [7] is a useful research direction.

References

[1] Anja Becker, Jean-Sébastien Coron, and Antoine Joux. Improved generic algorithms for hard knapsacks. In Kenneth G. Paterson, editor, *Advances in Cryptology – EUROCRYPT 2011*, pages 364–385, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[2] Anja Becker, Antoine Joux, Alexander May, and Alexander Meurer. Decoding random binary linear codes in 2n/20: How $1 + 1 = 0$ improves information set decoding. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, pages 520–536, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[3] Tim Beyne and Yu Long Chen. Information-theoretic security with asymmetries. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part IV*, volume 14923 of *Lecture Notes in Computer Science*, pages 463–494. Springer, 2024.

[4] Ritam Bhaumik, Avijit Dutta, Akiko Inoue, Tetsu Iwata, Ashwin Jha, Kazuhiko Minematsu, Mridul Nandi, Yu Sasaki, Meltem Sönmez Turan, and Stefano Tessaro. Cryptographic treatment of key control security - in light of NIST SP 800-108. In Yael Tauman Kalai and Seny F. Kamara, editors, *Advances in Cryptology - CRYPTO 2025 - 45th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2025, Proceedings, Part V*, volume 16004 of *Lecture Notes in Computer Science*, pages 371–403. Springer, 2025.

[5] Ritam Bhaumik, Avijit Dutta, Tetsu Iwata, Ashwin Jha, Kazuhiko Minematsu, Mridul Nandi, Yu Sasaki, Meltem Sönmez Turan, and Stefano Tessaro. A note on feedback-PRF mode of KDF from NIST SP 800-108. Cryptology ePrint Archive, Paper 2025/1586, 2025.

[6] Leif Both and Alexander May. Decoding linear codes with high error rate and its impact for lpn security. In Tanja Lange and Rainer Steinwandt, editors, *Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018, Fort Lauderdale, FL, USA*, volume 10786 of *Lecture Notes in Computer Science book series*, pages 25–46. Springer, 2018.

[7] Tanja Lange Daniel J. Bernstein and Christiane Peters. Smaller decoding exponents: ball-collision decoding. In *In Advances in Cryptology - CRYPTO 2011*, volume 6841 of *Lecture Notes in Computer Science book series*, pages 743–760. Springer.

[8] Ilya Dumer. On minimum distance decoding of linear codes. In *In Proceedings of the 5th Joint Soviet-Swedish International Workshop on Information Theory*, pages 50–52, 1991.

[9] Andre Esser and Floyd Zweydinger. New time-memory trade-offs for subset sum improving isd in theory and practice. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology - EUROCRYPT 2023- 42nd Annual International Conference on the Theory and Applications of Cryptographic*

Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part V, volume 14008 of *of Lecture Notes in Computer Science*, page 360–390. Springer, 2023.

[10] Nick Howgrave-Graham and Antoine Joux. New generic algorithms for hard knapsacks. In H. Gilbert, editor, *Annual International Conference on the Theory and Applications of Cryptographic Techniques EUROCRYPT 2010: Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science book series*. Springer, 2010.

[11] Ashwin Jha and Mridul Nandi. A survey on applications of h-technique: Revisiting security analysis of PRP and PRF. *Entropy*, 24(4):462, 2022.

[12] Pierre Karpman and Charlotte Lefevre. Time-memory tradeoffs for large-weight syndrome decoding in ternary codes. In Yohei Watanabe Goichiro Hanaoka, Junji Shikata, editor, *25th IACR International Conference on Practice and Theory of Public-Key Cryptography, Virtual Event, March 8–11, 2022, Proceedings, Part I*, volume 13177 of *Lecture Notes in Computer Science book series*. Public-Key Cryptography – PKC 2022, 2022.

[13] Charles Meyer-Hilfiger Kevin Carrier, Thomas Debris-Alazard and Jean-Pierre Tillich. Statistical decoding 2.0: Reducing decoding to lpn. In *In Advances in Cryptology - ASIACRYPT 2022*, volume 13794 of *Lecture Notes in Computer Science book series*, pages 477–507. Springer, 2022.

[14] Pil Joong Lee and Ernest F. Brickell. An observation on the security of mceliece's public-key cryptosystem. In Christoph G. Gunther, editor, *Advances in Cryptology - EUROCRYPT '88, Workshop on the Theory and Application of of Cryptographic Techniques, Davos, Switzer- land*, volume 330 of *Lecture Notes in Computer Science book series*, pages 275–280. Springer, 1988.

[15] Jeffrey S. Leon. A probabilistic algorithm for computing minimum weights of large error- correcting codes. *IEEE Trans. Inf. Theory*, 34(5):1354–1359, 1988.

[16] Alexander May, Alexander Meurer, and Enrico Thomae. Decoding random linear codes in $\tilde{O}(20.054n)$. In *Proceedings of the 17th International Conference on The Theory and Application of Cryptology and Information Security*, ASIACRYPT'11, page 107–124, Berlin, Heidelberg, 2011. Springer-Verlag.

[17] Alexander May and Ilya Ozerov. On computing nearest neighbors with applications to de- coding of binary linear codes. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria*, volume 9056 of *Lecture Notes in Computer Science book series*, pages 203–228. Springer, 2015.

[18] NIST. Calls for proposals, 2016. (accessed on 9th August, 2023).

[19] Eugene Prange. The use of information sets in decoding cyclic codes. *IRE Trans. Inf. Theory*, 8(5):5–9, 1962.

[20] Richard Schroeppel and Adi Shamir. A $t = \mathcal{O}(2^{n/2})$, $s = \mathcal{O}(2^{n/4})$ algorithm for certain np-complete problems. *SIAM Journal on Computing*, 10(3):456–464, 1981.

[21] Yaobin Shen, Lei Wang, and Dawu Gu. Security analysis of NIST key derivation using pseudorandom functions. Cryptology ePrint Archive, Paper 2025/815, 2025.

[22] Jacques Stern. A method for finding codewords of small weight. In Gérard D. Cohen and Jacques Wolfmann, editors, *Coding Theory and Applications, 3rd International Colloquium, Toulon, France, November 2-4, 1988, Proceedings*, volume 388 of *Lecture Notes in Computer Science*, page 106–113. Springer, 1988.

[23] David Wagner. A generalized birthday problem. In H. Gilbert, editor, *Advances in Cryptology — CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science book series*. Springer, 2002.