Identity-based Encryption from Codes with Rank Metric

Philippe Gaborit[1], Adrien Hauteville[1,2], Duong Hieu Phan[1], and Jean-Pierre Tillich[2]

[1] Université de Limoges, XLIM-DMI, 123, Av. Albert Thomas, 87060 Limoges Cedex, France
[2] Inria de Paris, 2 rue Simone Iff, CS 42112, 75589 Paris Cedex 12, France

Abstract. Code-based cryptography has a long history, almost as long as the history of public-key encryption (PKE). While we can construct almost all primitives from codes such as PKE, signature, group signature etc, it is a long standing open problem to construct an identity-based encryption from codes. We solve this problem by relying on codes with rank metric.

The concept of identity-based encryption (IBE), introduced by Shamir in 1984, allows the use of users' identifier information such as email as public key for encryption. There are two problems that makes the design of IBE extremely hard: the requirement that the public key can be an arbitrary string and the possibility to extract decryption keys from the public keys. In fact, it took nearly twenty years for the problem of designing an efficient method to implement an IBE to be solved. The known methods of designing IBE are based on different tools: from elliptic curve pairings by Sakai, Ohgishi and Kasahara and by Boneh and Franklin in 2000 and 2001 respectively; from the quadratic residue problem by Cocks in 2001; and finally from the Learning-with-Error problem by Gentry, Peikert, and Vaikuntanathan in 2008.

Among all candidates for post-quantum cryptography, there only exist thus lattice-based IBE. In this paper, we propose a new method, based on the hardness of learning problems with rank metric, to design the first code-based IBE scheme. In order to overcome the two above problems in designing an IBE scheme, we first construct a rank-based PKE, called RankPKE, where the public key space is dense and thus can be obtained from a hash of any identity. We then extract a decryption key from any public key by constructing an trapdoor function which relies on RankSign - a signature scheme from PQCrypto 2014.

In order to prove the security of our schemes, we introduced a new problem for rank metric: the Rank Support Learning problem (RSL). A high technical contribution of the paper is devoted to study in details the hardness of the RSL problem.

Keywords. Code-based cryptography, Rank metric, IBE, PKE.

1 Introduction

1.1 Code-based Cryptography

Code-based cryptography has a long history, which began by the McEliece cryptosystem in 1978, followed by the Niederreiter scheme in 1986. [38]. The difficult problem involved in these cryptosystems is the Syndrome Decoding problem, which consists in recovering from a random matrix $\boldsymbol{H}$ and from a syndrome $\boldsymbol{s} = \boldsymbol{H}\boldsymbol{e}^T$, the small (Hamming) weight error vector $\boldsymbol{e}$ associated to $\boldsymbol{s}$. The idea of these encryption schemes is to consider as public key a masking of a decodable code. Although this masking could be broken for some special families of codes like Reed-Solomon codes or Reed-Muller codes, the original family of binary Goppa codes proposed by McEliece in 1978 is still today considered as secure, and the indistinguishability of Goppa codes from random codes for standard encryption parameters remains unbroken. Few years later Alekhnovich proposed in 2003 [2] a cryptosystem relying on random instances of codes but with larger size of encrypted messages. Code-based cryptosystems had still very large public keys, but from the year 2005 [22], inspired by the NTRU approach, structured matrices, and in particular quasi-cyclic matrices, where also considered for public keys leading to cryptosystems with only a small hidden structure like for instance the MDPC cryptosystem of 2013 [37].

However, when signature schemes were already known for a long time in number theory based cryptography, finding a signature scheme (not based on the Fiat-Shamir heuristic) had been an open problem for quite some time, until the CFS scheme of Courtois, Finiasz and Sendrier in 2001 [15], the scheme is an hash-and-sign signature which computes a signature as a small (Hamming) weight vector associated to a random syndrome. Although this latter scheme has some advantages, like a short signature size, the small weight vector has a logarithmic weight in the length of the code, which implies a super polynomial complexity and very large public keys, which makes it difficult to use it for advanced encryption schemes like for instance identity-based encryption.

Beside systems based on the Hamming metric, cryptosystems relying on a different metric, the rank metric, were introduced in 1991 by Gabidulin *et al.*[21]. This system, which is an analogue of the McEliece cryptosystem but with a different metric was based on Gabidulin codes, which are analogue codes to Reed-Solomon codes for rank metric. These codes having a very strong structure, they were difficult to mask (as their Hamming counterpart the Reed-Solomon codes), and in practice all cryptosystems based on these codes were broken. Meanwhile the rank metric approach had a strong advantage over the Hamming approach, the fact that the generic decoding problems are inherently more difficult than for Hamming metric. In some sense the general decoding problems for rank metric are to Hamming metric, what is the discrete logarithm problem over the group of an elliptic elliptic curve rather than on the ring $\mathbb{Z}/p\mathbb{Z}$. Again, following the approach of NTRU and the (Hamming) MDPC cryptosystem, an analogue cryptosystem, was proposed in 2013 for rank metric: the Low Rank Parity Check (LRPC) cryptosystem [24], as its cousins the MDPC and the NTRU cryptosys-

tems, this system benefits from a poor structure which also seems (as for MDPC and NTRU) to limit the attacks to general attacks on the rank syndrome decoding problem.

In 2014, a new signature scheme, the RankSign scheme, based on LRPC codes was introduced by Gaborit *et al.* at PQCrypto 2014, [26]. This signature scheme is also a hash-and-sign signature scheme which inverts a random syndrome, but at the difference of the CFS scheme, the weight of the returned signature is linear in the length of the code, which implies smaller size of public key. Moreover beside its poor structure, inherited from the LRPC structure, the system comes with a security proof on information leaking from signatures. Thus we are eventually able to use this hash-and-sign signature scheme as a brick for the first IBE scheme based on coding theory.

1.2 Identity based Encryption

The notion of identity-based encryption (IBE) was introduced by Shamir [42]. This gives an alternative to the standard notion of public-key encryption. In an IBE scheme, the public key associated with a user can be an arbitrary identity string, such as his e-mail address, and others can send encrypted messages to a user using this arbitrary identity without having to rely on a public-key infrastructure.

The main technical difference between a public key encryption (PKE) and IBE is the way the public and private keys are bound and the way of verifying those keys. In a PKE scheme, verification is achieved through the use of a certificate which relies on a public-key infrastructure. In an IBE, there is no need of verification of the public key but the private key is managed by a Trusted Authority (TA).

Difficulty in designing an IBE. There are two main difficulties in designing an IBE in comparison with a PKE

1. In a PKE, one often generates a public key from a secret key and normally, well-formed public keys are exponentially sparse. In an IBE scheme, any identity should be mapped to a public key and there is no known technique to randomly generate a point in an exponentially sparse space. Regev's public key encryption is an example [40]. In order to circumvent this problem, Gentry *et. al.* proposed a "dual" of a public-key cryptosystem, in which public keys are first generated in a primal space such that they are *dense*: every point in the primal space corresponds to a public-key and thus via a random oracle, one can map any identity to a valid public key.
2. For some PKE, the public keys are dense and one can thus map any identity to a well-formed public key. However, the difficulty is to extract the corresponding secret key from the public key. ElGamal's public key encryption [17] is an example because from a public key y in a cyclic group generated by g, there is no trapdoor for the discrete log problem that allows to find the corresponding secret key x such that $g^x = y$. In order to circumvent this problem, bilinear maps have been used [10].

Beside the technical difficulties in the design, achieving security in IBE is much more complicated than in PKE. The main difference is that in IBE, except the challenge identity that the adversary aims to attack, any other identities can be corrupted. Therefore the simulator has to be able to generate secret keys for all identities but the challenge identity. Under the above difficulties in the design and in proving the security, it took nearly twenty years for finding efficient methods to implement IBE.

There are currently three classes of IBE schemes: from elliptic curve pairings introduced by Sakai, Ohgishi and Kasahara [41] and by Boneh and Franklin in [10]; from the quadratic residue problem by Cocks in 2001 [14]; and from hard problems on lattice by Gentry, Peikert, and Vaikuntanathan [29]. These pioneer works inspired then many other ideas to improve the efficiency or to strengthen the security, in particular to avoid the use of the random oracle. We can name some very interesting schemes in the standard model: pairing-based schemes [8,9,46,28,12,45] and lattice-based scheme [13,1,11]. It is still not known how to devise an IBE scheme from quadratic residue problem without random oracles. We explain below a new method to achieve the first IBE scheme in the coding theory, with the help of rank metric codes and in the random oracle model.

Achieving IBE in Euclidean Metric. Let us first recall the technique in lattice that helps to construct IBE schemes. One of the major breakthroughs in lattice cryptography was the work of Gentry, Peikert, and Vaikuntanathan [29], that showed how to use a short trapdoor basis to generate short lattice vectors without revealing the trapdoor. This was used to give the first lattice-based construction of a secure identity-based encryption scheme.

Let us start with Regev's scheme [40]. Associated to a matrix $\boldsymbol{A} \in \mathbb{Z}_q^{n\times m}$, one generates the public key as $\boldsymbol{p} = \boldsymbol{s}A + \boldsymbol{e}$ for $\boldsymbol{s} \in \mathbb{Z}_q^n$ and a short vector $\boldsymbol{e}$. The set of possible public keys are points near a lattice point and are thus exponentially sparse. Gentry, Peikert, and Vaikuntanathan introduced a dual version of the Regev's scheme in exchanging the role of public key and of secret key in defining the public key as $\boldsymbol{u} \stackrel{\text{def}}{=} A\boldsymbol{e}^T \mod q$ for short $\boldsymbol{e} \in \mathbb{Z}^m$. The public keys are now dense, any identity could be mapped via a random oracle to a point in $\mathbb{Z}_q^n$ which will then be used as the corresponding public key. The key property is, with a carefully designed trapdoor $\boldsymbol{T}$, from a random public key $\boldsymbol{u} \in \mathbb{Z}_q^n$, the preimage $\boldsymbol{e}$ of the function $f_{\boldsymbol{A}}(\boldsymbol{e}) := A\boldsymbol{e} \mod q$ can be sampled in a space of well-defined short vectors used as the secret keys.

Achieving IBE in Rank Metric: Our technique. It seems very hard to give a rank metric analogue version of the above lattice technique. The main reason is due to the difficulty of obtaining a robust analysis of such a presampling function. However, we can overcome this difficulty in another way which perfectly fits the rank metric. We still keep the public key as $\boldsymbol{p} = \boldsymbol{s}A + \boldsymbol{e}$ for $\boldsymbol{e}$ of low rank (say at most r) in $\mathbb{F}_{q^m}^n$, and for $\boldsymbol{A}$ and $\boldsymbol{s}$ drawn uniformly at random in $\mathbb{F}_{q^m}^{(n-k)\times n}$ and $\mathbb{F}_{q^m}^{n-k}$ respectively, where $\mathbb{F}_{q^m}$ is the finite field over q^m elements. The main feature of the rank metric which will be used in what follows is that we can

choose the bound r to be above the Gilbert Varshamov (RGV) bound for rank codes and this gives us two ingredients to design an IBE:

- with r carefully chosen above the RGV bound, we can still invert the function $f(\boldsymbol{s}, \boldsymbol{e}) = \boldsymbol{s}A + \boldsymbol{e}$. This relies on the RankSign system with a trapdoor to compute the pre-image of the function f [26].
- with overwhelming probability, any point $\boldsymbol{p}$ has a preimage $(\boldsymbol{s}, \boldsymbol{e})$ such that $f(\boldsymbol{s}, \boldsymbol{e}) = \boldsymbol{p}$. We can thus map an arbitrary identity to a valid public key $\boldsymbol{p}$, by using a random oracle as in the case of the GPV scheme.

Rank metric vs. Hamming and Euclidean Metric. Rank metric and Hamming metric are very different metrics. This difference reflects for instance in the size of balls: when the number of elements of a ball of radius r in the Hamming metric for $\{0,1\}^n$ is bounded above by 2^n, for rank metric the number of elements is exponential but with a quadratic exponent depending on r. In practice, it means that even if it is possible to construct a trapdoor function for the Hamming distance such as the CFS signature scheme [15], the dimension of the dual code used there has to be sublinear in its length, when for rank metric it is possible to obtain such a trapdoor function for constant rate codes. This latter property makes it very difficult to use such a trapdoor function for the Hamming distance in order to build an IBE scheme whereas it is tractable for the rank metric.

Moreover one strong advantage of rank metric is the potential size of public keys. If one considers the general syndrome decoding problem $\boldsymbol{H}\boldsymbol{x}^T = \boldsymbol{s}$ (for the hardest case), because of the complexity of the best known attacks for rank metric (see [25]), and for λ a security parameter, the size of $\boldsymbol{H}$ is in $\mathcal{O}\left(\lambda^{\frac{3}{2}}\right)$ for rank metric when it is in $\mathcal{O}\left(\lambda^2\right)$ for Hamming and Euclidean metrics.

1.3 Hardness of Problems in Rank Metric

The computational complexity of decoding $\mathbb{F}_{q^m}$-linear codes for rank metric has been an open question for almost 25 years since the first paper on rank based cryptography in 1991 [21]. Recently a probabilistic reduction to decoding in Hamming distance was given in [27]. On a practical complexity point of view the complexity of practical attacks grows very fast with the size of parameters, and there is a structural reason for this: for Hamming distance a key notion in the attacks is counting the number of words of length n and support size t, which corresponds to the notion of Newton binomial coefficient $\binom{n}{t}$, whose value is exponential and upper bounded by 2^n. In the case of rank metric, counting the number of possible supports of size r for a rank code of length n over $\mathbb{F}_{q^m}$ corresponds to counting the number of subspaces of dimension r in $\mathbb{F}_{q^m}$: the Gaussian binomial coefficient of size roughly $q^{r(m-r)}$, whose value is also exponential but with a quadratic term in the exponent.

1.4 Our Contribution

The contributions of this paper are two-fold:

On the cryptographic aspect: we design new cryptographic primitives based on the rank metric. The final objective is to design an IBE scheme, but on the way, we also introduce a new PKE scheme which perfectly fits a transformation from PKE to IBE. This shows a potential versatility of the use of rank metric in cryptography: it gives a credible alternative to Euclidean metric in the perspective of post-quantum cryptography and it has some advantages compared to Hamming metric as it is still a open question to construct an IBE scheme based on the Hamming metric. We emphasize that the design of an IBE scheme often opens the way to reach more advanced primitives such as Broadcast Encryption, Attribute-based Encryption and Functional Encryption.

On the algorithmic aspect: the security of the new constructions that we introduce relies on the hardness of three algorithmic problems. Two of them are well known problems, namely the Rank Syndrome Decoding Problem and the Augmented Low Rank Parity Check Code problem. However the last one is new and we call it the Rank Support Learning problem. A large part of the paper is devoted to study the hardness of the Rank Support Learning problem and more specifically

- we prove the equivalence between the Rank Support Learning problem and Rank Decoding with parity-check matrices defined over a subfield;
- we show that this problem can also be tackled by finding low weight-codewords in a certain code;
- we show that this problem can be viewed as the rank metric analogue of a rather old problem in the Hamming metric for which the best known algorithms are exponential;
- based on this analogy we give an algorithm of exponential complexity to handle this problem over the rank metric.

2 Background on rank metric and cryptography

2.1 Notation

In the whole article, q denotes a power of a prime p. The finite field with q elements is denoted by $\mathbb{F}_q$ and more generally for any positive integer m the finite field with q^m elements is denoted by $\mathbb{F}_{q^m}$. We will frequently view $\mathbb{F}_{q^m}$ as an m-dimensional vector space over $\mathbb{F}_q$.

We use bold lowercase and capital letters to denote vectors and matrices respectively. We will view vectors here either as column or row vectors. It will be clear from the context whether it is a column or a row vector. For two matrices $\boldsymbol{A}, \boldsymbol{B}$ of compatible dimensions, we let $(\boldsymbol{A}|\boldsymbol{B})$ and $\left(\frac{\boldsymbol{A}}{\boldsymbol{B}}\right)$ respectively denote the horizontal and vertical concatenations of $\boldsymbol{A}$ and $\boldsymbol{B}$.

If S is a finite set, $x \xleftarrow{\$} S$ denotes that x is chosen uniformly at random among S. If $\mathcal{D}$ is a distribution, $x \leftarrow \mathcal{D}$ denotes that x is chosen at random according to $\mathcal{D}$.

We also use the standard $\mathcal{O}()$, $\Omega()$ and $\Theta()$ notation and also the "soft-O" notation $\tilde{\mathcal{O}}(\ )$, where $f(x) = \tilde{\mathcal{O}}\big(g(x)\big)$ means that $f(x) = \mathcal{O}\big(g(x)\log(g(x))^k\big)$ for some k.

2.2 Definitions

In the whole article, the space $\mathbb{F}_{q^m}^n$ will be endowed with the following metric

Definition 1 (Rank metric over $\mathbb{F}_{q^m}^n$). *Let $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{F}_{q^m}^n$ and consider an arbitrary basis $(\beta_1, \ldots, \beta_m) \in \mathbb{F}_{q^m}^m$ of $\mathbb{F}_{q^m}$ viewed as an m-dimensional vector space over $\mathbb{F}_q$. We decompose each entry x_j in this basis $x_j = \sum_{i=1}^m m_{ij}\beta_i$. The $m \times n$ matrix associated to $\boldsymbol{x}$ is given by $\boldsymbol{M}(\boldsymbol{x}) = (m_{ij})_{\substack{1 \le i \le m \\ 1 \le j \le n}}$. The rank weight $\|\boldsymbol{x}\|$ of $\boldsymbol{x}$ is defined as*

$$\|\boldsymbol{x}\| \stackrel{def}{=} \operatorname{Rank} \boldsymbol{M}(\boldsymbol{x}).$$

The associated distance $\operatorname{rd}(\boldsymbol{x}, \boldsymbol{y})$ between elements $\boldsymbol{x}$ and $\boldsymbol{y}$ in $\mathbb{F}_{q^m}^n$ is defined by $\operatorname{rd}(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|$.

Remark 1. It is readily seen that this distance does not depend on the basis that is chosen. We refer to [36] for more details on the rank distance.

A rank code $\mathcal{C}$ of length n and dimension k over the field $\mathbb{F}_{q^m}$ is a subspace of dimension k of $\mathbb{F}_{q^m}^n$ embedded with the rank metric. The minimum rank distance of the code $\mathcal{C}$ is the minimum rank of non-zero vectors of the code. One also considers the usual inner product which allows to define the notion of dual code. An important notion which differs from the Hamming distance, is the notion of support. Let $\boldsymbol{x} = (x_1, x_2, \cdots, x_n) \in \mathbb{F}_{q^m}^n$ be a vector of rank weight r. We denote by $E \stackrel{\text{def}}{=} \langle x_1, x_2, \cdots, x_n \rangle_{\mathbb{F}_q}$ the $\mathbb{F}_q$-linear subspace of $\mathbb{F}_{q^m}$ generated by linear combinations over $\mathbb{F}_q$ of $x_1, x_2, \cdots, x_n$. The vector space E is called the **support** of $\boldsymbol{x}$ and is denoted by $\operatorname{Supp}(\boldsymbol{x})$. In the following, $\mathcal{C}$ is a rank metric code of length n and dimension k over $\mathbb{F}_{q^m}$. The matrix $\boldsymbol{G}$ denotes a $k \times n$ generator matrix of $\mathcal{C}$ and $\boldsymbol{H}$ is one of its parity check matrix.

Bounds for rank metric codes. The classical bounds for the Hamming metric have straightforward rank metric analogues, since two of them are of interest for the paper we recall them below.

Definition 2 (Rank Gilbert-Varshamov bound (RGV)). *The number of elements $S(n, m, q, t)$ of a sphere of radius t in $\mathbb{F}_{q^m}^n$, is equal to the number of $m \times n$ q-ary matrices of rank t. For $t = 0$ $S_0 = 1$, for $t \ge 1$ we have (see [36]):*

$$S(n, m, q, t) = \prod_{j=0}^{t-1} \frac{(q^n - q^j)(q^m - q^j)}{q^t - q^j}.$$

From this we deduce the volume $B(n,m,q,t)$ of a ball of radius t in $\mathbb{F}_{q^m}^n$ to be:

$$B(n,m,q,t) = \sum_{i=0}^{t} S(n,m,q,i).$$

In the linear case the Rank Gilbert-Varshamov bound $RGV(n,k,m,q)$ for an $[n,k]$ linear code over $\mathbb{F}_{q^m}$ is then defined as the smallest integer t such that $B(n,m,q,t) \geq q^{m(n-k)}$.

The Gilbert-Varshamov bound for a rank code $\mathcal{C}$ with parity-check matrix $\boldsymbol{H}$, corresponds to the smallest rank weight r for which, for any syndrome $\boldsymbol{s}$, there exists on average a word $\boldsymbol{e}$ of rank weight r such that $\boldsymbol{He} = \boldsymbol{s}$. To give an idea of the behavior of this bound, it can be shown that, asymptotically in the case $m = n$ ([36]): $\frac{RGV(n,k,m,q)}{n} \sim 1 - \sqrt{\frac{k}{n}}$.

Singleton bound. The classical Singleton bound for a linear $[n,k]$ rank code of minimum rank r over $\mathbb{F}_{q^m}$ works in the same way as for linear codes (by finding an information set) and reads $r \leq n - k + 1$: in the case when $n > m$ this bound can be rewritten as $r \leq 1 + \lfloor \frac{(n-k)m}{n} \rfloor$ [36].

2.3 Decoding rank codes

We will be interested in codes for the rank metric which can be efficiently decoded. At the difference of Hamming metric, there do not exist many families which admit an efficient decoding for large rank weight error (ideally we would like to go up to the RGV bound).

Deterministic decoding of rank codes. Essentially there is only one family of rank codes which can be decoded in a deterministic way: the Gabidulin codes [20]. These codes are an analogue of the Reed-Solomon codes where polynomials are replaced by q-polynomials and benefit from the same decoding properties (cf [20] for more properties on these codes). A Gabidulin code of length n and dimension k over $\mathbb{F}_{q^m}$ with $k \leq n \leq m$ can decode up to $\frac{n-k}{2}$ errors in a deterministic way.

Probabilistic decoding of rank codes. Besides the deterministic decoding of Gabidulin codes, which does not reach the RGV bound and hence is not optimal, it is possible to decode up to the RGV bound a simple family of codes. In this subsection we present the construction which allows with a probabilistic decoding algorithm to attain the RGV bound. These codes are adapted from codes in the subspace metric (a metric very close from the rank metric) which can be found in [43].

Definition 3 (Simple codes). *A code $\mathcal{C}$ is said to be (n,k,t)-simple (or just simple when t, k n are clear from the context), when it has a parity-check matrix $\boldsymbol{H}$ of the form*

$$\boldsymbol{H} = \left(\begin{array}{c|c} \boldsymbol{I}_{n-k} & \begin{array}{c} \mathbf{0_t} \\ \\ \boldsymbol{R} \end{array} \end{array} \right)$$

where $\boldsymbol{I}_{n-k}$ the $(n-k)\times(n-k)$ identity matrix, $\mathbf{0_t}$ is the zeromatrix of size $t\times k$ and $\boldsymbol{R}$ is a matrix over $\mathbb{F}_{q^m}$ of size $k\times(n-k-t)$. It is called a random simple code if $\boldsymbol{R}$ is chosen uniformly at random among matrices of this size.

Proposition 1. *Let $\mathcal{C}$ be a random (n,k,t)-simple code with $t < \frac{m+n-\sqrt{(m-n)^2+4km}}{2}$ and w an integer. If $w \leqslant t$ then $\mathcal{C}$ can decode an error of weight w with probability of failure $p_f \sim \frac{1}{q^{t-w+1}}$ when $q \to \infty$.*

The proof of this proposition is given in the appendix B.1.

The success of decoding depends essentially on the probability $1-p_f$ to recover the space E from the t first coordinates of $\boldsymbol{s}$, this probability can be made as small as needed by decoding less than t errors or by increasing q.

In term of complexity of decoding, one has just a system to invert in $(n-t)w$ unknowns in $\mathbb{F}_q$. Notice that the bound $\frac{m+n-\sqrt{(m-n)^2+4km}}{2}$ corresponds asymptotically to the Rank Gilbert-Varshamov bound. Thus a simple code can asymptotically decodes up to the Rank Gilbert-Varshamov bound with probability $1-\mathcal{O}\left(\frac{1}{q}\right)$. In the special case $m=n$ and $w=t\approx n\left(1-\sqrt{\frac{k}{n}}\right)$ (the Rank Gilbert-Varshamov bound), the system has $\mathcal{O}\left(n^2\right)$ unknowns, so the decoding complexity is bounded by $\mathcal{O}\left(n^6\right)$ operations in $\mathbb{F}_q$. This decoding algorithm is better than the Gabidulin code decoder in term of correction capability since it corrects up to $n\left(1-\sqrt{\frac{k}{n}}\right)$ errors when Gabidulin codes can not decode more than $\frac{n-k}{2}$ errors.

2.4 Difficult problem for rank-based cryptography

Rank-based cryptography generally relies on the hardness of syndrome decoding for the rank metric. It is defined as the well known syndrome decoding problem but here the Hamming metric is replaced by the rank metric.

Definition 4 (Rank (Metric) Syndrome Decoding Problem (RSD)). *Let $\boldsymbol{H}$ be a full-rank $(n-k)\times n$ matrix over $\mathbb{F}_{q^m}$ with $k\leq n$, $\boldsymbol{s}\in\mathbb{F}_{q^m}^{n-k}$ and w an integer. The problem is to find $\boldsymbol{x}\in\mathbb{F}_{q^m}^{n}$ such that $\mathrm{rank}(\boldsymbol{x})=w$ and $\boldsymbol{H}\boldsymbol{x}=\boldsymbol{s}$. We denote this problem as the $\mathsf{RSD}_{q,m,n,k,w}$ problem.*

The RSD problem has recently been proven hard in [27] on probabilistic reduction. This problem has an equivalent dual version. Let $\boldsymbol{H}$ be a parity-check matrix of a code $\mathcal{C}$ and $\boldsymbol{G}$ be a generator matrix. Then the RSD problem

is equivalent to find $\boldsymbol{m} \in \mathbb{F}_{q^m}^k$ and $\boldsymbol{x} \in \mathbb{F}_{q^m}^n$ such that $\boldsymbol{mG} + \boldsymbol{x} = \boldsymbol{y}$ with $\mathrm{Rank}\, \boldsymbol{x} = r$ and $\boldsymbol{y}$ some preimage of $\boldsymbol{s}$ by $\boldsymbol{H}$. We can now give the decisional version of this problem:

Definition 5 (Decisional Rank Syndrome Decoding Problem (DRSD)). *Let $\boldsymbol{G}$ be a full-rank $k \times n$ matrix over $\mathbb{F}_{q^m}$, $\boldsymbol{m} \in \mathbb{F}_{q^m}^k$ and $\boldsymbol{x} \in \mathbb{F}_{q^m}^n$ of weight r. Can we distinguish the pair $(\boldsymbol{G}, \boldsymbol{mG} + \boldsymbol{x})$ from $(\boldsymbol{G}, \boldsymbol{y})$ with $\boldsymbol{y} \xleftarrow{\$} \mathbb{F}_{q^m}^n$?*

The same problem in the Hamming metric Decisional Syndrome Decoding problem (DSD), viewed as an LPN problem with a fixed number of samples (which is equivalent to the syndrome decoding problem), is proven hard in [3] with a reduction to the syndrome decoding problem for the Hamming metric. We can use the same technique as in [23,27] to prove that DRSD is hard in the worst case. The general idea is that a distinguisher D_R with non negligible advantage for DRSD problem can be used to construct another distinguisher D for DSD with a non negligible advantage. The complete proof can be found in the appendix B.2.

2.5 Complexity of the rank decoding problem

As explained earlier in the introduction the complexity of practical attacks grows very fast with the size of parameters, there exist two types of generic attacks on the problem:

Combinatorial attacks: these attacks are usually the best ones for small values of q (typically $q = 2$) and when n and k are not too small; when q increases, the combinatorial aspect makes them less efficient. The best attacks generalize the basic information set decoding approach in a rank metric context. Interestingly enough, the more recent improvements based on birthday paradox do not seem to generalize in rank metric because of the different notion of support.

In practice, when $m \leqslant n$, the best combinatorial attacks have complexity $\mathcal{O}\big((n-k)^3 m^3 q^{(r-1)\lfloor \frac{(k+1)m}{n} \rfloor}\big)$ [25].

Algebraic attacks: the particular nature of rank metric makes it a natural field for algebraic attacks and solving by Groebner basis, since these attacks are largely independent of the value of q and in some cases may also be largely independent on m. These attacks are usually the most efficient ones when q increases. There exist different types of algebraic modeling which can then be solved with Groebner basis techniques ([35], [19], [18], [25]). Algebraic attacks usually consider algebraic systems on the base field $\mathbb{F}_q$, it implies that the number of unknowns is quadratic in the length of the code. Since the general complexity of Groebner basis attacks is exponential in the number of unknowns, it induces for cryptographic parameters, general attacks with a quadratic exponent in the length of the code, as for combinatorial attacks.

3 A New Public Key Encryption

3.1 Public-Key Encryption

Let us briefly remind that a public-key encryption scheme $\mathcal{S}$ is defined by three algorithms: the key generation algorithm KeyGen which, on input the security parameter, produces a pair of matching public and private keys (pk, sk); the encryption algorithm $\mathsf{Enc}_{pk}(m; r)$ which outputs a ciphertext c corresponding to the plaintext $m \in \mathcal{M}$, using random coins $r \in \mathcal{R}$; and the decryption algorithm $\mathsf{Dec}_{sk}(c)$ which outputs the plaintext m associated to the ciphertext c.

It is now well-admitted to require *semantic security* (*a.k.a. polynomial security* or *indistinguishability of encryptions* [30], denoted IND): if the attacker has some *a priori* information about the plaintext, it should not learn more with the view of the ciphertext. More formally, this security notion requires the computational indistinguishability between two messages, chosen by the adversary, one of which has been encrypted. The issue is to find which one has been actually encrypted with a probability significantly better than one half. More precisely, we define the advantage $\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{S}}(\mathcal{A})$, where the adversary $\mathcal{A}$ is seen as a 2-stage Turing machine $(\mathcal{A}_1, \mathcal{A}_2)$ by

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{S}}(\mathcal{A}) \stackrel{\text{def}}{=} 2 \times \Pr\left[\begin{array}{l}(pk, sk) \leftarrow \mathsf{KeyGen}, (m_0, m_1, s) \leftarrow \mathcal{A}_1(pk), \\ b \stackrel{R}{\leftarrow} \{0,1\}, c = \mathsf{Enc}_{pk}(m_b) : \mathcal{A}_2(m_0, m_1, s, c) = b\end{array}\right] - 1.$$

This advantage should ideally be a negligible function of the security parameter.

3.2 Description of the cryptosystem RankPKE

First, we need to define what we call a homogeneous matrix which will be used in encryption.

Definition 6 (Homogeneous Matrix). *A matrix* $\boldsymbol{M} = (m_{ij})_{\substack{1 \le i \le a \\ 1 \le j \le b}} \in \mathbb{F}_{q^m}^{a \times b}$ *is homogeneous of weight* d *if all its coefficients belong to the same* $\mathbb{F}_q$*-vector subspace of dimension* d*, that is to say*

$$\dim_{\mathbb{F}_q} \langle m_{ij} \rangle = d$$

We now introduce a public-key encryption, called $\mathsf{RankPKE}$. Let $\boldsymbol{A}$ be drawn uniformly at random in $\mathbb{F}_{q^m}^{(n-k)\times n}$. We need it to be of full rank. This happens with overhelming probability (i.e. $1 - \mathcal{O}\big(q^{-m(k+1)}\big)$). Let W_r be the set of all the words of rank r and of length n , i.e. $W_r = \{\boldsymbol{x} \in \mathbb{F}_{q^m}^n : \|\boldsymbol{x}\| = r\}$. The system $\mathsf{RankPKE}$ works as follows:

- $\mathsf{RankPKE.KeyGen}$:
  - generate $\boldsymbol{A} \stackrel{\$}{\leftarrow} \mathbb{F}_{q^m}^{(n-k)\times n}$
  - generate $\boldsymbol{s} \stackrel{\$}{\leftarrow} \mathbb{F}_{q^m}^{n-k}$ and $\boldsymbol{e} \stackrel{\$}{\leftarrow} W_r$
  - compute $\boldsymbol{p} = \boldsymbol{s}\boldsymbol{A} + \boldsymbol{e}$

- define $\boldsymbol{G} \in \mathbb{F}_{q^m}^{k' \times n'}$ a generator matrix of a public code $\mathcal{C}$ which can decode (efficiently) errors of weight up to wr, where w is defined just below.
- define $sk = \boldsymbol{s}$ and $pk = (\boldsymbol{A}, \boldsymbol{p}, \boldsymbol{G})$

– $\mathsf{RankPKE.Enc}((\boldsymbol{A}, \boldsymbol{p}, \boldsymbol{G}), \boldsymbol{m})$:
Let $\boldsymbol{m} \in \mathbb{F}_{q^m}^{k'}$ be the message we want to encrypt. We generate a random homogeneous matrix $\boldsymbol{U} \in \mathbb{F}_{q^m}^{n \times n'}$ of weight w. Then we can compute the ciphertext $(\boldsymbol{C}, \boldsymbol{x})$ of $\boldsymbol{m}$ as :

$$\left(\begin{array}{c} \boldsymbol{A} \\ \hline \boldsymbol{p} \end{array}\right) \boldsymbol{U} + \left(\begin{array}{c} \boldsymbol{0} \\ \hline \boldsymbol{mG} \end{array}\right) = \left(\begin{array}{c} \boldsymbol{C} \\ \hline \boldsymbol{x} \end{array}\right)$$

– $\mathsf{RankPKE.Dec}(\boldsymbol{s}, (\boldsymbol{C}, \boldsymbol{x}))$:
- use the secret key $\boldsymbol{s}$ to compute:

$$\begin{aligned} \left(\begin{array}{cc} \boldsymbol{s} & |-1 \end{array}\right) \left(\begin{array}{c} \boldsymbol{C} \\ \hline \boldsymbol{x} \end{array}\right) &= \boldsymbol{sC} - \boldsymbol{x} = \boldsymbol{sAU} - \boldsymbol{pU} - \boldsymbol{mG} \\ &= \boldsymbol{sAU} - (\boldsymbol{sA} + \boldsymbol{e})\boldsymbol{U} - \boldsymbol{mG} = -\boldsymbol{eU} - \boldsymbol{mG} \end{aligned}$$

- since $\boldsymbol{U}$ is homogeneous, we have $\|\boldsymbol{eU}\| \leqslant wr$. Therefore, by using the decoding algorithm of $\mathcal{C}$, we recover $\boldsymbol{m}$.

The expansion rate of this cryptosystem is $\frac{n-k+1}{R}$ where $R = \frac{k'}{n'}$ is the rate of $\mathcal{C}$.

3.3 Security

Definition 7 (Rank Support Learning (RSL)). *Let $\boldsymbol{A}$ be a random full-rank matrix of size $(n-k) \times n$ over $\mathbb{F}_{q^m}$ and V be a subspace of $\mathbb{F}_{q^m}$ of dimension w. Let $\mathcal{O}$ be an oracle which gives samples of the form $(\boldsymbol{A}, \boldsymbol{Au})$, where $\boldsymbol{u} \xleftarrow{\$} V^n$. The $\mathsf{RSL}_{q,m,n,k,w}$ problem is to recover V given only access to the oracle.*
We say that the problem is (N, t, ε)-hard if for every probabilistic algorithm $\mathcal{A}$ running in time t, we have

$$\mathrm{Prob}[\mathcal{A}(\boldsymbol{A}, \boldsymbol{AU}) = V] \leqslant \varepsilon, \quad \boldsymbol{U} \xleftarrow{\$} V^{n \times N}$$

When we want to stress the fact that we care about the problem where we are allowed to make exactly N calls to the oracle, we denote this the $\mathsf{RSL}_{q,m,n,k,w,N}$ problem. The pair $(\boldsymbol{A}, \boldsymbol{AU})$ is refered to as an instance of the $\mathsf{RSL}_{q,m,n,k,w,N}$ problem. The corresponding decisional problem, namely DRSL, is to distinguish $(\boldsymbol{A}, \boldsymbol{AU})$ from $(\boldsymbol{A}, \boldsymbol{Y})$ where $\boldsymbol{Y} \xleftarrow{\$} \mathbb{F}_{q^m}^{(n-k) \times N}$.

Proposition 2. *The $\mathsf{RSL}_{q,m,n,k,w,N}$ is as hard as the $\mathsf{RSD}_{q,m,n,k,w}$ problem.*

Proof. Let $\boldsymbol{A}$ be a full-rank $(n-k)\times n$ matrix over $\mathbb{F}_{q^m}$ and $\boldsymbol{x}\in\mathbb{F}_{q^m}^{n-k}$ of rank w. Let $\boldsymbol{s}=\boldsymbol{A}\boldsymbol{x}$. $(\boldsymbol{A},\boldsymbol{s})$ is an instance of the $\mathsf{RSD}_{q,m,n,k,w}$ problem.
Let $\boldsymbol{S}$ be a matrix obtained by the concatenation of N times the vector $\boldsymbol{s}$. $(\boldsymbol{A},\boldsymbol{S})$ is an instance of the $\mathsf{RSL}_{q,m,n,k,w,N}$ problem.
If we are able to solve any instances of the $\mathsf{RSL}_{q,m,n,k,w,N}$ problem, then we can recover the support of $\boldsymbol{x}$ and solve the instance $(\boldsymbol{A},\boldsymbol{s})$.
We can use this technique to solve any instances of the $\mathsf{RSD}_{q,m,n,k,w}$ problem, which proves that the $\mathsf{RSL}_{q,m,n,k,w,N}$ is as hard as the $\mathsf{RSD}_{q,m,n,k,w}$ problem in the worst case.

Security of the DRSL and DRSD problems.
We have already seen in the previous section that the DRSD problem is hard. As for other problems in cryptography (like DDH [7,16]), the best known attacks on the $\mathsf{DRSL}_{q,m,n,k,w,N}$ problem consist in solving the same instance of the $\mathsf{RSL}_{q,m,n,k,w,N}$ problem, so we make the assumption that the $\mathsf{DRSL}_{q,m,n,k,w,N}$ problem is difficult.

Theorem 1. *Under the assumption that* DRSL *is hard, the scheme* RankPKE *is semantically secure.*

Proof. We proceed by a sequence of games.

Game $\mathbf{G}_0$: This is the real IND-CPA attack game. The RankPKE.KeyGen is run and then, a 2-stage poly-time adversary $\mathcal{A}=(\mathcal{A}_1,\mathcal{A}_2)$ is fed with the public key $pk=(\boldsymbol{A},\boldsymbol{p},\boldsymbol{G}')$. Then, $\mathcal{A}_1$ outputs a pair of messages $(\boldsymbol{m}_0,\boldsymbol{m}_1)$. Next a challenge ciphertext is produced by flipping a coin b and producing a ciphertext $c^\star:=(C^\star,\boldsymbol{x}^\star)$ of $\boldsymbol{m}^\star=\boldsymbol{m}_b$.

This ciphertext $c^\star$ comes from a random homogeneous matrix $\boldsymbol{U}\in\mathbb{F}_{q^m}^{n\times n'}$ of weight w and then $c^\star=\mathsf{RankPKE.Enc}((\boldsymbol{A},\boldsymbol{p},\boldsymbol{G}'),\boldsymbol{m}_b)$. On input $c^\star$, $\mathcal{A}_2$ outputs bit b'. We denote by S_0 the event $b'=b$ and use the same notation S_n in any game $\mathbf{G}_n$ below.

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{RankPKE}}(\mathcal{A})=\mid 2\Pr[\mathsf{S}_0]-1\mid$$

Game $\mathbf{G}_1$: In this game, we replace $\boldsymbol{p}=\boldsymbol{s}A+\boldsymbol{e}$ in RankPKE.KeyGen by $\boldsymbol{p}\xleftarrow{\$}\mathbb{F}_{q^m}^n$. Under the hardness of the DRSD problem, the two games $\mathbf{G}_1$ and $\mathbf{G}_0$ are indistinguishable:

$$\mid\Pr[\mathsf{S}_1]-\Pr[\mathsf{S}_0]\mid\leq\epsilon_{\mathsf{drsd}},$$

where ϵ_{drsd} is the bound on the successful probability of the attacks against the problem DRSD.

Game $\mathbf{G}_2$: In this game, we replace $(C^\star,\boldsymbol{x}^\star)$ in $\mathbf{G}_1$ by $(C^\star\xleftarrow{\$}\mathbb{F}_{q^m}^{(n-k)\times n'},x^\star\xleftarrow{\$}\mathbb{F}_{q^m}^{n'})$.

As $\boldsymbol{x}^\star$ is perfectly random, $\boldsymbol{x}^\star-\boldsymbol{m}^\star\boldsymbol{G}$ is also perfectly random. In other words, this game replaces $\begin{pmatrix}\boldsymbol{A}\\\boldsymbol{p}\end{pmatrix}\boldsymbol{U}=\begin{pmatrix}C^\star\\\boldsymbol{x}^\star-\boldsymbol{m}^\star\boldsymbol{G}\end{pmatrix}$ by a perfectly random matrix. Therefore, the indistinguishability of the two games $\mathbf{G}_2$ and $\mathbf{G}_1$ follows from the

hardness of the DRSL problem, applying it to the matrix $\boldsymbol{A}' = \begin{pmatrix} \boldsymbol{A} \\ \boldsymbol{p} \end{pmatrix}$ which is perfectly random because $\boldsymbol{A}$ and $\boldsymbol{p}$ are both perfectly random. Thus

$$|\Pr[\mathsf{S}_2] - \Pr[\mathsf{S}_1]| \leq \epsilon_{\mathsf{drsl}},$$

where ϵ_{drsl} is the bound on the successful probability of the attacks against the DRSL problem.

Advantage Zero. In this last game, as the ciphertext challenge $(C^\star, \boldsymbol{x}^\star)$ is perfectly random, b is perfectly hidden to any adversary $\mathcal{A}$.

$$|\Pr[\mathsf{S}_2]| = \frac{1}{2}$$

4 On the difficulty of the rank support learning problem

The purpose of this section is to give some evidence towards the difficulty of the support learning problem $\mathsf{RSL}_{q,m,n,k,w,N}$ by

- explaining that it is the rank metric analogue of a problem in Hamming metric (the so called support learning problem) which has already been useful to devise signature schemes and for which after almost twenty years of existence only algorithms of exponential complexity are known;
- explaining that it is a problem which is provably hard for $N = 1$ and that it becomes easy only for very large values of N;
- giving an algorithm which is the analogue in the rank metric of the best known algorithm for the support learning problem which is of exponential complexity. This complexity is basically smaller by a multiplicative factor which is only of order $q^{-\beta N}$ (for some $\beta < 1$) than the complexity of solving the rank syndrome decoding problem $\mathsf{RSD}_{q,m,n,k,w}$;
- relating this problem to finding a codeword of rank weight w in a code where there are q^N codewords of this weight. It is reasonable to conjecture that the complexity of finding such a codeword gets reduced by a multiplicative factor which is at most q^N compared to the complexity of finding a codeword of rank weight w in a random code of the same length and dimension which has a single codeword of this weight;
- showing that this problem can also rephrased in terms of decoding a random code but defined over a larger alphabet ($\mathbb{F}_{q^{mN}}$ instead of $\mathbb{F}_{q^m}$).

4.1 A related problem : the support learning problem

The rank support learning problem can be viewed as the rank metric analogue of the support learning problem which can be expressed as follows.

Problem 1 (Support Learning). Let $\boldsymbol{A}$ be a random full-rank matrix of size $(n-k) \times n$ over $\mathbb{F}_q$ and I be a subset of $\{1, \ldots, n\}$ of size w. Let V be the subspace of $\mathbb{F}_q^n$ of vectors with support I, that is the set of vectors $\boldsymbol{u} = (u_i)_{1 \leq i \leq n} \in \mathbb{F}_q^n$ such that $u_i = 0$ when $i \notin I$. Let $\mathcal{O}$ be an oracle which gives samples of the form $(\boldsymbol{A}, \boldsymbol{Au})$, where $\boldsymbol{u} \xleftarrow{\$} V$. The support learning problem is to recover I given only access to the oracle.
We say that the problem is (N, t, ε)-hard if for every probabilistic algorithm $\mathcal{A}$ running in time t, we have

$$\text{Prob}[\mathcal{A}(\boldsymbol{A}, \boldsymbol{AU}) = V] \leqslant \varepsilon, \quad \boldsymbol{U} \xleftarrow{\$} V^N$$

When we want to stress the fact that we care about the problem where we are allowed to make exactly N calls to the oracle, we denote this the $\mathsf{SL}_{q,n,k,w,N}$ problem. The pair $(\boldsymbol{A}, \boldsymbol{AU})$ is referred to as an instance of the $\mathsf{SL}_{q,n,k,w,N}$ problem.

When $N = 1$ this is just the usual decoding problem of a random linear code with parity check matrix $\boldsymbol{A}$. In this case, the problem is known to be NP-complete [5]. When N is greater than 1, this can be viewed as a decoding problem where we are given N syndromes of N errors which have a support included in the same set I. This support learning problem with $N > 1$ has already been considered before in [33]. Its presumed hardness for moderate values of N was used there to devise a signature scheme [33], the so called KKS-scheme. Mounting a key attack without knowing any signature that has been computed for this key really amounts to solve this rank support learning problem even it was not stated exactly like this in the article. However, when we have signatures originating from this scheme, the problem is of a different nature. Indeed, it was found out in [**?**] that signatures leak information. The authors showed there that if we know M signatures, then we are given $\boldsymbol{A}$, $\boldsymbol{AU}$ but also M vectors in $\mathbb{F}_q^n$, $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_M$ whose support is included in I. The knowledge of those auxiliary $\boldsymbol{v}_i$'s help a great deal to recover I : it suffices to compute the union of their support which is very likely to reveal the whole set I. When the $\boldsymbol{v}_i$'s are random vectors in $\mathbb{F}_q^n$ of support included in I it is clearly enough to have a logarithmic number of them (in the size of the support I) to recover I. However this does not undermine the security of the support learning problem and just shows that the KKS-signature scheme is at best a one-time signature scheme.

Some progress on the support learning problem itself was achieved almost fifteen years later in [39]. Roughly speaking the idea there is to consider a code that has q^N codewords of weight at most w which correspond to all possible linear combinations of the $\boldsymbol{u}_i$'s and to use generic decoding algorithms of linear codes (which can also be used as low-weight codewords search algorithms) to recover one of those linear combinations. The process can then be iterated to reveal the whole support I. The fact that there are q^N codewords of weight $\leq w$ that are potential solutions for the low weight codeword search algorithm implies that we may expect to gain a factor of order q^N in the complexity of the algorithm when compared to finding a codeword of weight w in a random linear code which has

a single codeword of weight w. Actually the gain is less than this in practice. This seems to be due to the fact that we have highly correlated codewords (their support is for instance included in I). However, still there is some exponential speedup when compared to the single codeword case. This allowed to break all the parameters proposed in [33,34] but also those of [4] which actually relied on the same problem. However, as has been acknowledged in [39], this does not give a polynomial time algorithm for the support learning problem, it just gives an exponential speedup when compared to solving a decoding problem with an error of weight w. The parameters of the KKS scheme can easily be chosen in order to thwart this attack.

4.2 Both problems reduce to linear algebra when N is large enough

As explained before when $N = 1$ the support learning problem is NP-complete. The rank support learning problem is also hard in this case since it is equivalent to decoding in the rank metric an $\mathbb{F}_{q^m}$-linear code for which there is a randomized reduction to the NP-complete decoding problem in the Hamming metric [27]. It is also clear that both problems become easy when N is large enough and for the same reason : they basically amount to compute a basis of a linear space.

In the Hamming metric, this corresponds to the case when $N = w$. Indeed in this case, notice that the dimension of the subspace V is w. When the $\boldsymbol{u}_i$'s are generated randomly with support included in I they have a constant probability $K(q)$ (which is increasing with q and bigger than 0.288 in the binary case) to generate the space $\boldsymbol{A}V$. Once we know this space, the problem becomes easy. Indeed let $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n$ be the canonical generators of $\mathbb{F}_q^n$ (i.e. $\boldsymbol{e}_i$ has only one non-zero entry which is its i-th entry that is equal to 1). We recover I by checking for all positions i in $\{1, \ldots, n\}$ whether $\boldsymbol{A}\boldsymbol{e}_i$ belongs to $\boldsymbol{A}V$ or not. If it is the case, then i belongs to I, if this is not the case, i does not belong to I.

There is a similar algorithm for the rank support learning problem. This should not come as a surprise since supports of code positions for the Hamming metric really correspond to subspaces of $\mathbb{F}_{q^m}$ for the rank metric metric as has been put forward in [25] (see also [32] for more details about this). The difference being however that we need much bigger values of N to mount a similar attack to the Hamming metric case. Indeed what really counts here is the space that can be generated by the $\boldsymbol{A}\boldsymbol{u}_i$'s where the $\boldsymbol{u}_i$'s are the columns of $\boldsymbol{U}$. It is nothing but the space $\boldsymbol{A}V^n$. Let us denote this space by W. This space is not $\mathbb{F}_{q^m}$-linear, however it is $\mathbb{F}_q$-linear and it is of dimension nw viewed as an $\mathbb{F}_q$-linear subspace of $\mathbb{F}_{q^m}^n$. When $N = nw$ we can mount a similar attack, namely we compute the space generated by linear combinations over $\mathbb{F}_q$ of $\boldsymbol{A}\boldsymbol{u}_1, \ldots, \boldsymbol{A}\boldsymbol{u}_{nw}$. They generate W with constant probability $K(q)$. When we look all $\mathbb{F}_q$-linear subspaces V' of $\mathbb{F}_{q^m}$ of dimension 1 (there are less than q^m of them) and check whether the subspace W' of dimension n given by $\boldsymbol{A}V'^n$ is included in $W = \boldsymbol{A}V^n$ or not. By taking the sum of the spaces for which this is the case we recover V. Actually the complexity of this algorithm can be improved by using in a more clever way the knowledge of W, but this is beyond the scope of this article and this algorithm is just here to explain the deep similarities between both cases

and to convey some intuition about when the rank support learning problem becomes easy.

This discussion raises the issue whether there is an algorithm "interpolating" standard decoding algorithms when $N = 1$ and linear algebra when $N = w$ in the Hamming metric case and $N = nw$ in the rank metric case. This is in essence what has been achieved in [39] for the Hamming metric and what we will do now here for the rank metric.

4.3 Solving the subspace problem with information-set decoding

There are two ingredients in the algorithm for solving the support learning problem in [39]. The first one is to set up an equivalent problem which amounts to find a codeword of weight $\leq w$ in a code which has q^N codewords of this weight. The second one is to use standard information set decoding techniques to solve this task and to show that it behaves better than in the case where there is up to a multiplicative constant a single codeword of this weight in the code. We are going to follow the same route here for the rank metric.

We begin by introducing the following $\mathbb{F}_q$-linear code

$$\mathcal{C} \stackrel{\text{def}}{=} \{\boldsymbol{x} \in \mathbb{F}_{q^m}^n : \boldsymbol{A}\boldsymbol{x} \in W_U\}$$

where W_U is the $\mathbb{F}_q$-linear subspace of $\mathbb{F}_{q^m}^{n-k}$ generated by linear combinations of the form $\sum_i \alpha_i \boldsymbol{A}\boldsymbol{u}_i$ where α_i belongs to $\mathbb{F}_q$ and the $\boldsymbol{u}_i$'s are the N column vectors forming the matrix $\boldsymbol{U}$. This code has the following properties.

Lemma 1. *Let $\mathcal{C}' \stackrel{def}{=} \{\sum_i \alpha_i \boldsymbol{u}_i : \alpha_i \in \mathbb{F}_q\}$. We have*

1. $\dim_{\mathbb{F}_q} \mathcal{C} \leq km + N$
2. $\mathcal{C}' \subset \mathcal{C}$
3. *all the elements of $\mathcal{C}'$ are of rank weight $\leq w$.*

[25] gives several algorithms for decoding $\mathbb{F}_{q^m}$-linear codes for the rank metric. The first one can be generalized in a straightforward way to codes which are just $\mathbb{F}_q$-linear as explained in more detail in [32]. This article also explains how this algorithm can be used in a straightforward way to search for low rank codewords in such a code. Here our task is to look for codewords of rank $\leq w$ which are very likely to lie in $\mathcal{C}'$ which would reveal a linear combination $\boldsymbol{c} = \sum_i \alpha_i \boldsymbol{u}_i$. This reveals in general V when $\boldsymbol{c}$ is of rank weight w simply by computing the vector space over $\mathbb{F}_q$ generated by the entries of $\boldsymbol{c}$. When the rank of $\boldsymbol{c}$ is smaller this yields a subspace of V and we will discuss later on how we finish the attack.

Let us concentrate now on analyzing how the first decoding algorithm of [25] behaves when we use it to find codewords of $\mathcal{C}$ of rank $\leq w$. For this, we have to recall how the support attack of [25] works.

We assume that we want to find a codeword of weight w in an $\mathbb{F}_q$-linear code which is a $\mathbb{F}_q$-subspace of $\mathbb{F}_{q^m}$ of dimension K. For the purpose of this algorithm, a codeword $\boldsymbol{c} = (c_1, \ldots, c_n) \in \mathbb{F}_{q^m}^n$ is also viewed as a matrix $(c_{ij})_{\substack{1 \leq i \leq m \\ 1 \leq j \leq n}}$ over $\mathbb{F}_q$

by writing the c_i's in a arbitrary $\mathbb{F}_q$ basis $(\beta_1, \ldots, \beta_m) \in \mathbb{F}_{q^m}^m$ of $\mathbb{F}_{q^m}$ viewed as vector space over $\mathbb{F}_q$: $c_i = \sum_{j=1}^m c_{ij}\beta_j$. There are $nm - K$ linear equations which specify the code that are satisfied by the c_{ij}'s of the form

$$\sum_{1 \leq i,j \leq m} h_{ij}^s c_{ij} = 0 \tag{1}$$

for $s = 1, \ldots, mn - K$. Algorithm 1 explains how a codeword of weight $\leq w$ is produced by the approach of [25]. The point of choosing r like this in this algorithm, i.e.

$$r \stackrel{\text{def}}{=} m - \left\lceil \frac{K}{n} \right\rceil \tag{2}$$

is that r is the smallest integer for which the linear system (3) has more equations than unknowns (and we therefore expect that it has generally only the all-zero solution).

Theorem 2. *Assume that* $w \leq \min\left(\left\lfloor \frac{K}{n} \right\rfloor, \left\lfloor \frac{N}{n} \right\rfloor + 1\right)$ *and that* $\frac{w + \left\lfloor \frac{K}{n} \right\rfloor}{2} \geq \left\lfloor \frac{N}{n} \right\rfloor$. *Let*

$$\begin{aligned} e_- &= \left(w - \left\lfloor \frac{N}{n} \right\rfloor\right)\left(\left\lfloor \frac{K}{n} \right\rfloor - \left\lfloor \frac{N}{n} \right\rfloor\right) \\ e_+ &= \left(w - \left\lfloor \frac{N}{n} \right\rfloor - 1\right)\left(\left\lfloor \frac{K}{n} \right\rfloor - \left\lfloor \frac{N}{n} \right\rfloor - 1\right) + n\left(\left\lfloor \frac{N}{n} \right\rfloor + 1\right) - N \end{aligned}$$

Algorithm 1 outputs an element of $\mathcal{C}'$ *with complexity* $\tilde{\mathcal{O}}\left(q^{\min(e_-, e_+)}\right)$. *We give the complete proof of this theorem in the appendix C.*

Remark 2.

1. When N and $K = km + N$ are multiple of n, say $N = \delta n$ and $K = \alpha R n + \delta$ (with $\alpha \stackrel{\text{def}}{=} \frac{m}{n}$, $R = \frac{k}{n}$) the complexity above simplifies to $\tilde{\mathcal{O}}\left(q^{\alpha R n(w-\delta)}\right)$. In other words the complexity gets reduced by a factor $q^{\alpha R \delta n} = q^{\alpha R N}$ when compared to finding a codeword of weight w in a random $\mathbb{F}_q$-linear code of the same dimension and length.
2. This approach is really suited to the case $m \leq n$. When $m > n$ we obtain better complexities by working on the transposed code (see [32] for more details about this approach).

Algorithm 1 algorithm that outputs a codeword of weight $\leq w$.

$r \leftarrow m - \lceil \frac{K}{n} \rceil$
loop
  $W \leftarrow$ random $\mathbb{F}_q$-subspace of dimension r of $\mathbb{F}_{q^m}$
  Compute a basis $\boldsymbol{f}^1 = (f_i^1)_{1 \leq i \leq m}, \ldots, \boldsymbol{f}^r = (f_i^r)_{1 \leq i \leq m}$ of W
  Make the assumption that the entries c_j of $\boldsymbol{c}$ can be written in the $\boldsymbol{f}^1, \ldots, \boldsymbol{f}^r$ basis as

$$c_j = \sum_{l=1}^{r} x_{lj} \boldsymbol{f}^l$$

  Rewrite the linear equations (1) by writing $c_{ij} = \sum_{l=1}^{r} x_{lj} f_i^l$
  to obtain $mn - K$ equations of the form

$$\sum_{1 \leq i,j \leq m} h_{ij}^s \sum_{l=1}^{r} x_{lj} f_i^l = 0 \tag{3}$$

  Define $(x_{ij})_{\substack{1 \leq i \leq r \\ 1 \leq j \leq n}}$ by $c_{ij} = \sum_{l=1}^{r} x_{lj} f_i^l$
  Solve this system (in the x_{ij}'s)
  if this system has a non zero solution **then**
    if $(\sum_{l=1}^{r} x_{lj} \boldsymbol{f}^l)_{1 \leq j \leq n}$ has rank weight $\leq w$ **then**
      return $(\sum_{l=1}^{r} x_{lj} \boldsymbol{f}^l)_{1 \leq j \leq n}$
    end if
  end if
end loop

4.4 Link between rank support learning and decoding over the rank metric

We have exploited here that for solving the rank support learning problem, it can be rephrased in terms of finding a codeword of low rank weight in a code that has many codewords of such low rank weight (namely the code $\mathcal{C}$ that has been introduced in this section). $\mathcal{C}$ is not a random code however, it is formed by a random subcode, namely the code $\mathcal{C}_0 = \{\boldsymbol{x} \in \mathbb{F}_{q^m}^n : \boldsymbol{A}\boldsymbol{x} = 0\}$ plus some non random part, namely $\mathcal{C}'$ which contains precisely the low rank codeword we are after. In other words $\mathcal{C}$ decomposes as

$$\mathcal{C} = \mathcal{C}_0 \oplus \mathcal{C}'$$

where $\mathcal{C}_0$ is a truly random code and $\mathcal{C}'$ is a subcode of $\mathcal{C}$ that contains the codewords of $\mathcal{C}$ of low-rank. $\mathcal{C}$ is therefore not really a random code.

There is a way however to rephrase the rank support learning problem as a problem of decoding a *random* code. The trick is to change the alphabet of the code. We define the code $\mathcal{C}_N$ as

$$\mathcal{C}_N = \{\boldsymbol{x} \in \mathbb{F}_{q^{mN}} : \boldsymbol{A}\boldsymbol{x} = 0\}.$$

In other words, $\mathcal{C}_N$ is a code defined over the extension field $\mathbb{F}_{q^{mN}}$ but with a random parity-check matrix with entries defined over $\mathbb{F}_{q^m}$.

There are several ways to equip $\mathbb{F}_{q^{mN}}^n$ with a rank metric. One of them consists in writing the entries c_i of a codeword $\boldsymbol{c} = (c_1, \ldots, c_n) \in \mathbb{F}_{q^{mN}}^n$ of $\mathcal{C}_N$ as column vectors $(c_{ij})_{1 \leq j \leq mN} \in \mathbb{F}_q^{mN}$ by expressing the entry c_i in a $\mathbb{F}_q$ basis of $\mathbb{F}_{q^{mN}}$ $(\beta_1, \ldots, \beta_{mN})$, i.e. $c_i = \sum_{1 \leq j \leq mN} c_{ij}\beta_j$ and replacing each entry by the corresponding vector to obtain an $mN \times n$ matrix. The rank of this matrix would then define the rank weight of a codeword. However, since $\mathbb{F}_{q^{mN}}$ is an extension field of $\mathbb{F}_{q^m}$ there are also other ways to define a rank metric. We will choose the following one here. First we decompose each entry c_i in an $\mathbb{F}_{q^m}$-basis $(\gamma_1, \ldots, \gamma_N)$ of $\mathbb{F}_{q^{mN}}$:

$$c_i = \sum_{j=1}^{N} \alpha_{(i-1)N+j}\gamma_j$$

where the α_i's belong to $\mathbb{F}_{q^m}$. The rank weight of $(c_1, \ldots, c_n)$ is then defined as the rank weight of the vector $(\alpha_i)_{1 \leq i \leq nN} \in \mathbb{F}_{q^m}^{nN}$ where the rank weight of the last vector is defined as we have done up to here, namely by replacing each entry α_i by a column vector $(\alpha_{ij})_{1 \leq j \leq m}$ obtained by taking the coordinates of α_i in some $\mathbb{F}_q$-basis of $\mathbb{F}_{q^m}$. In other words, the rank weight of $(c_1, \ldots, c_n)$ is defined as the rank of the associated $m \times nN$ matrix.

Let us now introduce the rank decoding problem with random parity check matrices defined over a smaller field.

Definition 8 (Rank Decoding with parity-check matrices defined over a subfield (RDPCSF)). *Let $\boldsymbol{A}$ be a random full-rank matrix of size $(n-k) \times n$ over $\mathbb{F}_{q^m}$ and $\boldsymbol{e} \in \mathbb{F}_{q^{mN}}^n$ be a random word of rank weight w. The $\mathsf{RDPCSF}_{q,m,n,k,w,N}$ problem is to recover $\boldsymbol{e}$ from the knowledge of $\boldsymbol{A} \in \mathbb{F}_{q^m}^{(n-k)\times n}$ and $\boldsymbol{A}\boldsymbol{e} \in \mathbb{F}_{q^{mN}}^{n-k}$.*

It turns out that the support learning problem and the rank decoding problem with parity-check matrices defined over a smaller field are equivalent

Theorem 3. *The problems $\mathsf{RSL}_{q,m,n,w,N}$ and $\mathsf{RDPCSF}_{q,m,n,w,N}$ are equivalent : any randomized algorithm solving one of this problem with probability $\geq \epsilon$ in time t can be turned into an algorithm for the other problem solving it with probability $\geq \epsilon$ in time $t + P(q, m, n, w, N)$, where P is a polynomial function of its entries.*

Proof. Let us consider an instance $(\boldsymbol{A}, \boldsymbol{AU})$ of the $\mathsf{RSL}_{q,m,n,w,N}$ problem. Denote the j-th column of $\boldsymbol{U}$ by $\boldsymbol{u}_j$. Define now $\boldsymbol{e} \in \mathbb{F}_{q^{mN}}^n$ by $\boldsymbol{e} = \sum_{j=1}^{N} \gamma_j \boldsymbol{u}_j$, where $(\gamma_1, \ldots, \gamma_N)$ is some $\mathbb{F}_{q^m}$-basis of $\mathbb{F}_{q^{mN}}$. From the definition of the rank weight we have chosen over $\mathbb{F}_{q^{mN}}^n$, it is clear that the rank weight of $\boldsymbol{e}$ is less than or equal to w. The pair $(A, \sum_{j=1}^{N} \gamma_j \boldsymbol{A}\boldsymbol{u}_j)$ is then an instance of the $\mathsf{RDPCSF}_{q,m,n,w,N}$ problem. It is now straightforward to check that we transform in this way a uniformly distributed instance of the $\mathsf{RSL}_{q,m,n,w,N}$ problem into a uniformly distributed instance of the $\mathsf{RDPCSF}_{q,m,n,w,N}$ problem. The aforementioned claim on the equivalence of the two problems follows immediately from this and the fact that when we know the space generated by the entries of the $\boldsymbol{u}_j$'s, we just have to solve a linear system to recover a solution of the decoding problem (this accounts for the additive polynomial overhead in the complexity).

Note that this reduction of the rank support learning problem to the problem of decoding a linear code over an extension field $\mathbb{F}_{q^{mN}}$ defined from a random parity-check matrix defined over the base field $\mathbb{F}_{q^m}$ works also for the Hamming metric : the support learning problem $\mathsf{SL}_{q,n,w,N}$ also reduces to decoding a linear code over an extension field $\mathbb{F}_{q^{mN}}$ defined from a random parity-check matrix defined over the base field $\mathbb{F}_{q^m}$ but this time for the Hamming metric over $\mathbb{F}_{q^{mN}}^n$. All these considerations point towards the same direction, namely that when N is not too large, the rank support learning problem should be a hard problem. It is for instance tempting to conjecture that this problem can not be solved q^N faster than decoding errors of rank weight w for an $[n, k]$ random linear code over $\mathbb{F}_{q^m}$. A similar conjecture could be made for the support learning problem.

5 Identity Based Encryption

Identity-based encryption schemes. An identity-based encryption (IBE) scheme is a tuple of algorithms $\mathsf{IBE} = (\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{Enc}, \mathsf{Dec})$ providing the following functionality. The trusted authority runs Setup to generate a master key pair (mpk, msk). It publishes the master public key mpk and keeps the master secret key msk private. When a user with identity ID wishes to become part of the system, the trusted authority generates a user decryption key $d_{ID} \xleftarrow{\$} \mathsf{KeyDer}(msk, ID)$, and sends this key over a secure and authenticated channel to the user. To send an encrypted message m to the user with identity ID, the sender computes the ciphertext $C \xleftarrow{\$} \mathsf{Enc}(mpk, ID, m)$, which can be decrypted by the user as $m \leftarrow \mathsf{Dec}(d_{ID}, C)$. We refer to [10] for details on the security definitions for IBE schemes.

Security. We define the security of IBE schemes through a game with an adversary. In the first phase, the adversary is run on input of the master public key of a freshly generated key pair $(mpk, msk) \xleftarrow{\$} \mathsf{Setup}$. In a chosen-plaintext attack ($\mathsf{IND}-\mathsf{CPA}$), the adversary is given access to a key derivation oracle $\mathcal{O}$ that on input an identity $ID \in \{0,1\}^*$ returns $d_{ID} \xleftarrow{\$} \mathsf{KeyDer}(msk, ID)$. At the end of the first phase, the adversary outputs two equal-length challenge messages $m_0, m_1 \in \{0,1\}^*$ and a challenge identity $ID \in \{0,1\}^\star$. The adversary is given a challenge ciphertext $C \xleftarrow{\$} \mathsf{Enc}(mpk, ID, m_b)$ for a randomly chosen bit b, and is given access to the same oracle $\mathcal{O}$ as during the first phase of the attack. The second phase ends when the adversary outputs a bit b'. The adversary is said to win the $\mathsf{IND}-\mathsf{CPA}$ game if $b' = b$ and if it never queried the key derivation oracle for the keys of any identity that matches the target identity.

Definition 9. *An* IBE *scheme is* $\mathsf{IND}-\mathsf{CPA}$*-secure if any poly-time adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *making at most a polynomial number of queries to the key derivation oracle, only has a negligible advantage in the* $\mathsf{IND}-\mathsf{CPA}$ *game described above,* i.e., *the following advantage is negligible:*

$$2 \times \Pr\left[\begin{array}{l}(mpk, msk) \xleftarrow{\$} \mathsf{Setup}, (ID, m_0, m_1, s) \leftarrow \mathcal{A}_1^{\mathcal{O}}(mpk), \\ b \xleftarrow{\$} \{0,1\}, c = \mathsf{Enc}(mpk, ID, (m_b)) : \mathcal{A}_2^{\mathcal{O}}(m_0, m_1, s, c) = b\end{array}\right] - 1.$$

5.1 Trapdoor Functions From RankSign

We now adapt the RankSign system to construct a trapdoor function, which is sufficient to convert our PKE to an IBE. Associated to a matrix $\boldsymbol{A} \in \mathbb{F}_{q^m}^{(n-k)\times n}$, we define the function $f_{\boldsymbol{A}}$ as follows:

$$\begin{aligned} f_{\boldsymbol{A}} : \mathbb{F}_{q^m}^{n-k} \times \mathbb{F}_{q^m}^{n} &\to \mathbb{F}_{q^m}^{n} \\ (\boldsymbol{s}, \boldsymbol{e}) &\mapsto \boldsymbol{s}\boldsymbol{A} + \boldsymbol{e} \end{aligned}$$

The matrix $\boldsymbol{A}$ will be generated with a trapdoor $\boldsymbol{T}$ such that $f_{\boldsymbol{A}}$ is a trapdoor function: from a random $\boldsymbol{p} \in \mathbb{F}_{q^m}^{n}$, with the trapdoor $\boldsymbol{T}$, one can sample $(\boldsymbol{s}, \boldsymbol{e}) = f_{\boldsymbol{A}}^{-1}(\boldsymbol{p})$ such that $\boldsymbol{e}$ is indistinguishable from a random element in W_r, the set of all the words of rank r and of length n, as defined in RankPKE. These properties will be sufficient for us to construct an IBE and reduce its security to the security of RankPKE. We now describe how we can get such a trapdoor function by relying on the RankSign system [26].

RankSign RankSign is a signature scheme based on the rank metric. Like other signature schemes based on coding theory [15], RankSign needs a family of codes with an efficient decoding algorithm. It takes on input a random word of the syndrome space (obtained from the hash of the file we want to sign) and outputs a word of small weight with the given syndrome. This is an instance of the RSD problem, with the difference that the matrix $\boldsymbol{H}$ has a trapdoor which makes the problem easy. The public key is a description of the code which hides its structure and the secret key, on the contrary, reveals the structure of the code, which allows the signer to solve the RSD problem. RankSign does not compute a codeword of weight below the Gilbert-Varshamov bound, but instead a codeword of weight r between the Gilbert-Varshamov bound and the Singleton bound. The idea is to use a family of the augmented Low Rank Parity Check codes (denoted $LRPC^+$), and an adapted decoding algorithm (called the General Errors/Erasures Decoding algorithm) to produce such a codeword from any syndrome. The decoding algorithm is probabilistic and the parameters of the code have to be chosen precisely in order to have a probability of success very close to 1. We refer to [26] for a complete description of the decoding algorithm and the signature algorithm.

Definition 10 (Augmented Low Rank Parity Check Codes). *Let $\boldsymbol{H}$ be an $\mathbb{F}_{q^m}^{(n-k)\times n}$ homogeneous matrix of full-rank and of weight d and $\boldsymbol{R} \in \mathbb{F}_{q^m}^{(n-k)\times t}$ be a random matrix. Let $\boldsymbol{P} \in GL_{n-k}(\mathbb{F}_{q^m})$ and $\boldsymbol{Q} \in GL_{n+t}(\mathbb{F}_q)$ be two invertible matrices (remark that the coefficients of $\boldsymbol{Q}$ belong to the base field). Let $\boldsymbol{H}' = \boldsymbol{P}(\boldsymbol{R}|\boldsymbol{H})\boldsymbol{Q}$ be the parity-check matrix of a code $\mathcal{C}$ of type $[n+t, t+k]$. By definition, such a code is an $LRPC^+$ code. If $t = 0$, $\mathcal{C}$ is an LRPC code.*

The public key of RankSign is the matrix $\boldsymbol{H}'$, the secret key is the structured matrix $(\boldsymbol{R}|\boldsymbol{H})$ and the trapdoor is the pair of matrices $(\boldsymbol{P}, \boldsymbol{Q})$.

We can now describe the trapdoor function $f_{\boldsymbol{A}}^{-1}$. Let $\boldsymbol{p} \in \mathbb{F}_{q^m}^{n+t}$ and $\boldsymbol{H}'$ the public key of an instance of RankSign. We choose $\boldsymbol{A}$ as a generator matrix of

a code with parity-check matrix $\boldsymbol{H}'$, i.e. as a full-rank matrix over $\mathbb{F}_{q^m}$ of size $(k+t)\times(n+t)$ which is such that $\boldsymbol{H}'\boldsymbol{A}^T = 0$. First, we compute $\boldsymbol{H}'\boldsymbol{p}$ and then we apply RankSign with trapdoor T to this syndrome to obtain a vector $\boldsymbol{e}$ of weight r such that $\boldsymbol{H}'\boldsymbol{p}^T = \boldsymbol{H}'\boldsymbol{e}^T$. Finally, we solve the linear system $\boldsymbol{s}\boldsymbol{A} = \boldsymbol{p}-\boldsymbol{e}$ of unknown $\boldsymbol{s}$ and the secret key associated to $\boldsymbol{p}$ is set to be $\boldsymbol{s}$. The security of the RankSign system is based on the assumption that $\boldsymbol{H}'$ is computationally indistinguishable from a random matrix.

Definition 11 (LRPC^+ problem[26]). *Given an augmented LRPC code, distinguish it from a random code with the same parameters.*

The hardness of this problem is studied in [26]. Currently the best attacks consist in recovering the structure of the LRPC by looking for small-weight words in the code, and the best algorithms for that are generic algorithms whose complexity is exponential [32].

Proposition 3. *Let $\boldsymbol{H}'$ be a public RankSign matrix and $\boldsymbol{A}$ be a generator matrix of the associated code. The two following distributions are computationally indistinguishable:*

Let $\mathcal{D}_0$ the distribution $(\boldsymbol{p},\boldsymbol{s},\boldsymbol{e})$ where $\boldsymbol{p} \xleftarrow{\$} \mathbb{F}_{q^m}^{n+t}$, $\boldsymbol{e} \in W_r$ is sampled from RingSign Algorithm such that $\boldsymbol{H}'\boldsymbol{e}^T = \boldsymbol{H}'\boldsymbol{p}^T$ and $\boldsymbol{s}$ is the solution of the linear system $\boldsymbol{x}\boldsymbol{A} = \boldsymbol{p}-\boldsymbol{e}$ of unknown $\boldsymbol{x}$.

Let $\mathcal{D}_1$ be the distribution $(\boldsymbol{p}',\boldsymbol{s}',\boldsymbol{e}')$ with $\boldsymbol{s}' \xleftarrow{\$} \mathbb{F}_{q^m}^{k+t}$, $\boldsymbol{e}' \xleftarrow{\$} W_r$ and $\boldsymbol{p}' = \boldsymbol{s}'\boldsymbol{A}+\boldsymbol{e}'$.

Precisely, the maximum advantage ϵ of the adversaries to distinguish $\mathcal{D}_0$ et $\mathcal{D}_1$ is bounded by: $\epsilon \leq \frac{2}{q} + \epsilon_{\mathsf{drsd}}$

Proof. Let $\mathcal{D}_2$ be the distribution $(\boldsymbol{s},\boldsymbol{e})$ where $\boldsymbol{s} \xleftarrow{\$} \mathbb{F}_{q^m}^{n-k}$ and $\boldsymbol{e}$ is a signature of $\boldsymbol{s}$ by RankSign with the public key $\boldsymbol{H}'$ (*i.e,* $\|\boldsymbol{e}\| = r$ and $\boldsymbol{H}'\boldsymbol{e}^T = \boldsymbol{s}$). Let $\mathcal{D}_3$ be the distribution $(\boldsymbol{H}'\boldsymbol{e}'^T, \boldsymbol{e}'^T)$ with $\boldsymbol{e}' \xleftarrow{\$} W_r$.

According to the proof of Theorem 2 of [26], a sample $(\boldsymbol{H}'\boldsymbol{e}'^T, \boldsymbol{e}'^T) \leftarrow \mathcal{D}_3$ is distributed exactly as $\mathcal{D}_2$ except if $(\boldsymbol{H}'\boldsymbol{e}'^T, \boldsymbol{e}'^T)$ is not T-decodable and the probability that the latter occurs is less than $\frac{2}{q}$. Therefore an adversary can not distinguish $\mathcal{D}_2$ from $\mathcal{D}_3$ with an advantage larger than $\frac{2}{q}$.

Now, we can prove the proposition. First, let us examine the distribution $\mathcal{D}_0$. Since $\boldsymbol{H}'$ is a linear map and $\boldsymbol{p} \xleftarrow{\$} \mathbb{F}_{q^m}^{n+t}$, $\boldsymbol{s} = \boldsymbol{H}'\boldsymbol{p}^T$ is uniformly distributed among $\mathbb{F}_{q^m}^{n-k}$. This implies $(\boldsymbol{\sigma},\boldsymbol{e}) \leftarrow \mathcal{D}_2$. Moreover, $\boldsymbol{p}-\boldsymbol{e}$ is uniformly distributed among the words of the code generated by $\boldsymbol{A}$, hence $\boldsymbol{s} \xleftarrow{\$} \mathbb{F}_{q^m}^{k+t}$.

According to the indistinguishability of $\mathcal{D}_2$ and $\mathcal{D}_3$, the distribution of $\boldsymbol{e}'$ and $\boldsymbol{e}$ are computationally indistinguishable. $\boldsymbol{s}'$ and $\boldsymbol{s}$ are both uniformly distributed. Finally, based on the assumption that the DRSD problem is hard, $\boldsymbol{p}'$ and $\boldsymbol{p}$ are indistinguishable.

Summing up these two steps, the advantage of an adversary to distinguish $\mathcal{D}_0$ from $\mathcal{D}_1$ is bounded by $\frac{2}{q} + \epsilon_{\mathsf{drsd}}$. □

5.2 Scheme

Our IBE system uses a random oracle H which maps the identity into the public keys space $\mathbb{F}_{q^m}^{n+t}$ of our encryption scheme.

- $\mathsf{IBE.Setup}$
  - choose the parameters (n, m, k, d, t) of the scheme according to RankSign. The secret master key is the triplet of matrices $\boldsymbol{P}, (\boldsymbol{R}|\boldsymbol{H})$ and $\boldsymbol{Q}$ such that $\boldsymbol{H}$ is a parity-check matrix of an $[n, k]$ LRPC code of weight d over $\mathbb{F}_{q^m}$, $\boldsymbol{R} \xleftarrow{\$} \mathbb{F}_{q^m}^{(n-k)\times t}$, $\boldsymbol{P} \xleftarrow{\$} GL_{n-k}(\mathbb{F}_{q^m})$ and $\boldsymbol{Q} \xleftarrow{\$} GL_{n+t}(\mathbb{F}_q)$. Let $\boldsymbol{A}$ be a full rank $(k+t) \times (n+t)$ matrix over $\mathbb{F}_{q^m}$ such $\boldsymbol{H}'\boldsymbol{A}^T = 0$ with $\boldsymbol{H}' = \boldsymbol{P}(\boldsymbol{R}|\boldsymbol{H})\boldsymbol{Q}$ and the trapdoor $\boldsymbol{T}$ is $(\boldsymbol{P}, \boldsymbol{Q})$.
  - define $\boldsymbol{G} \in \mathbb{F}_{q^m}^{k'\times n'}$ to be a generator matrix of a public code $\mathcal{C}'$ which can decode (efficiently) errors of weight up to wr as in $\mathsf{RankPKE.KeyGen}$.
  - return $mpk = (\boldsymbol{A}, \boldsymbol{G})$ and $msk = \boldsymbol{T}$
- $\mathsf{IBE.KeyDer}(\boldsymbol{A}, \boldsymbol{T}, id)$:
  - compute $\boldsymbol{p} = H(id)$
  - compute $(\boldsymbol{s}, \boldsymbol{e}) = f_{\boldsymbol{A}}^{-1}(\boldsymbol{p})$ by using the trapdoor $\boldsymbol{T}$
  - store $(id, \boldsymbol{s})$ and return $\boldsymbol{s}$
- $\mathsf{IBE.Enc}(id, \boldsymbol{m})$:
  - compute $\boldsymbol{p} = H(id)$
  - return $\boldsymbol{c} = \mathsf{RankPKE.Enc}((\boldsymbol{A}, \boldsymbol{p}, \boldsymbol{G}), \boldsymbol{m})$
- $\mathsf{IBE.Dec}(\boldsymbol{s}, \boldsymbol{c})$: return $\mathsf{RankPKE.Dec}(\boldsymbol{s}, \boldsymbol{c})$.

5.3 Security

We now state the security of the IBE system.

Theorem 4. *Under the assumption that the* LRPC^+ *problem is hard and the* RankPKE *is secure, the* IBE *system described above is* $\mathsf{IND-CPA}$*-secure in the random oracle model:*

$$\epsilon_{\mathsf{ibe}} \leq \frac{2q_H}{q} + \epsilon_{\mathsf{lrpc}^+} + q_H(\epsilon_{\mathsf{drsd}} + \epsilon_{\mathsf{pke}})^{3}$$

where $\epsilon_{\mathsf{lrpc}^+}, \epsilon_{\mathsf{pke}}, \epsilon_{\mathsf{ibe}}$ *are respectively the bound on the advantage of the attacks against the* LRPC^+ *problem, the* RankPKE *system and the* IBE *system, and* q_H *is the maximum number of distinct hash queries to* H *that an adversary can make.*

[3] As in the lattice-based IBE scheme of Gentry, Peikert, and Vaikuntanathan [29], we lose a factor q_H in the reduction from PKE to IBE. Moreover, because of the lack of a statistical indistinguishability in the preimage sampling as in [29], we also lose an additional cost of $\frac{2q_H}{q} + q_H\epsilon_{\mathsf{drsd}}$ which require us to use a large q. Fortunatly, the efficiency of our scheme is $\mathcal{O}\big(\log q\big)$.

Proof. We proceed by a sequence of games.

Game $\mathbf{G}_0$: This is the real IND-CPA attack game. The $\mathsf{IBE.Setup}$ is run and then, a 2-stage poly-time adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ is fed with the public key $mpk = (\boldsymbol{A}, \boldsymbol{G})$. $\mathcal{A}_1$ can ask queries to H and key queries. Then, $\mathcal{A}_1$ outputs a challenge identity $id^\star$, which is different from the key queries $\mathcal{A}_1$ already asked, and a pair of messages $(\boldsymbol{m}_0, \boldsymbol{m}_1)$. Next a challenge ciphertext is produced by flipping a coin b and producing a ciphertext $c^\star = \mathsf{IBE.Enc}(id^\star, \boldsymbol{m}_b)$.

On input $c^\star$, $\mathcal{A}_2$ can continue to ask queries to H and key queries which are different from $id^\star$, and finally outputs bit b'. We denote by S_0 the event $b' = b$ and use the same notation S_n in any game $\mathbf{G}_n$ below.

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{IBE}}(\mathcal{A}) = \mid 2\Pr[\mathsf{S}_0] - 1 \mid$$

We assume without loss of generality that, for any identity id that $\mathcal{A}$ wants to corrupt, $\mathcal{A}$ already queried H on id. In particular, we can assume that $\mathcal{A}$ will query the challenge identity $id^\star$ to H.

As this is the real attack game, for a key query on an identity id, the $\mathsf{IBE.KeyDer}(\boldsymbol{A}, \boldsymbol{T}, id)$ is run and the secret key is given to $\mathcal{A}$. We recall this algorithm:

- compute $\boldsymbol{p} = H(id)$
- compute $(\boldsymbol{s}, \boldsymbol{e}) = f_{\boldsymbol{A}}^{-1}(\boldsymbol{p})$ by using the trapdoor $\boldsymbol{T}$:
  - compute $\boldsymbol{H}'\boldsymbol{p}$ and then we apply RankSign with trapdoor $\boldsymbol{T}$ to this syndrome to obtain a vector $\boldsymbol{e}$ of weight r such that $\boldsymbol{H}'\boldsymbol{p} = \boldsymbol{H}'\boldsymbol{e}$.
  - solve the linear system $\boldsymbol{s}\boldsymbol{A} = \boldsymbol{p} - \boldsymbol{e}$ of unknown $\boldsymbol{s}$ and the secret key associated to $\boldsymbol{p}$ is set to be $\boldsymbol{s}$.
- store $(id, \boldsymbol{s})$ and return $\boldsymbol{s}$.

Game $\mathbf{G}_1$: In this game, we modify the answers to the key queries so that it does not require the trapdoor $\boldsymbol{T}$ anymore. In order to make the answers coherent, we also need to simulate the queries to the hash queries to H. We maintain a list List_H, initially set to empty, to store the tuples $(id, \boldsymbol{p}, \boldsymbol{s})$ where $\boldsymbol{p}$ is the value that we respond to the H query on id, and $\boldsymbol{s}$ is the secret key which corresponds to the public key $\boldsymbol{p}$ we generate. The simulation is precised in the following way:

- Hash queries: on $\mathcal{A}$'s jth distinct query id_j to H:
  - randomly choose a vector $\boldsymbol{e}_j$ of weight r
  - randomly choose $\boldsymbol{s}_j$
  - define $\boldsymbol{p}_j = H(id) = \boldsymbol{s}_j\boldsymbol{A} + \boldsymbol{e}_j$
  - add the tuple $(id_j, \boldsymbol{p}_j, \boldsymbol{s}_j)$ to List_H and return $\boldsymbol{p}_j$ to $\mathcal{A}$.
- Secret key queries: when $\mathcal{A}$ asks for a secret key for the identity id, we retrieve the tuple $(id, \boldsymbol{p}, \boldsymbol{s})$ from the List_H and return $\boldsymbol{s}$ to $\mathcal{A}$.

Now, looking back at the Proposition 3, we remark that the set of q_H samples $(\boldsymbol{p}_j, \boldsymbol{s}_j, \boldsymbol{e}_j)$ in the previous game come from the distribution $\mathcal{D}_0^{q_H}$ and the set of q_H samples $(\boldsymbol{p}_j, \boldsymbol{s}_j, \boldsymbol{e}_j)$ in this game come from the distribution $\mathcal{D}_1^{q_H}$. We thus have:

$$|\Pr[\mathsf{S}_1] - \Pr[\mathsf{S}_0]| \leq \frac{2q_H}{q} + q_H \epsilon_{\mathsf{drsd}}$$

Game $\mathbf{G}_2$: As the objective is to reduce the security of the IBE to the security of RankPKE, in this game, we define the matrix $\boldsymbol{A}$ to be a random matrix as in the RankPKE. Because the simulation in the previous game does not use the trapdoor $\boldsymbol{T}$, we can keep the simulation for hash queries and key queries exactly unchanged. By the assumption that the LRPC^+ problem is hard, this game is indistinguishable from the previous game:

$$|\Pr[\mathsf{S}_2] - \Pr[\mathsf{S}_1]| \leq \epsilon_{\mathsf{lrpc}^+}$$

Game $\mathbf{G}_3$: We can now reduce the security of the IBE in the previous game to the security of RankPKE. We are given the public key $\boldsymbol{p}^\star$ of RankPKE and try to break the semantic security of RankPKE. Intuitively, we proceed as follows. We will try to embed the given public key $\boldsymbol{p}^\star$ of RankPKE to $H(id^\star)$. The IBE for $id^\star$ becomes thus a RankPKE with the same distribution of public keys. We can then use the given challenge ciphertext of RankPKE as the challenge ciphertext to $\mathcal{A}$ and whenever $\mathcal{A}$ can break IBE, we can break RankPKE. The difficulty in this strategy is that we should correctly guess the challenge identity $id^\star$. In a selective game where $\mathcal{A}$ has to announce $id^\star$ at the beginning of the game, we know this identity. However, in the adaptive game that we consider, we need make a guess on the challenge identity among all the identities queried to H. This explains why we lose a factor q_H in the advantage to attack RankPKE.

Now, formally, on input a random matrix $\boldsymbol{A}$ and a public key $\boldsymbol{p}^\star$ for the RankPKE, we choose an index i among $1, \dots q_H$ uniformly at random and change the answer for the ith query to H and for the challenge as follows:

- Hash queries: on $\mathcal{A}$'s jth distinct query id_j to H: if $j = i$, then add the tuple $(id_j, \boldsymbol{p}^\star, \perp)$ to List_H and return $\boldsymbol{p}^\star$ to $\mathcal{A}$. Otherwise for $j \neq i$, do the same as in the previous game.
- Secret key queries: when $\mathcal{A}$ asks for a secret key for the identity id, retrieve the tuple $(id, \boldsymbol{p}, \boldsymbol{s})$ from the List_H. If $\boldsymbol{s} \neq \perp$, return $\boldsymbol{s}$ to $\mathcal{A}$, otherwise output a random bit and abort.
- Challenge ciphertext: when $\mathcal{A}$ submits a challenge identity $id^\star$, different from all its secret key queries, and two messages m_0, m_1, if $id^\star = id_i$, i.e., $(id^\star, \boldsymbol{p}^\star, \perp) \notin \mathsf{List}_H$, then output a random bit and abort. Otherwise, we also submits the messages m_0, m_1 to the challenger and receive a challenge ciphertext $c^\star$. We return then $c^\star$ to $\mathcal{A}$.

When $\mathcal{A}$ terminates and returns a bit b, we also outputs b. We now analyze the advantage to break RankPKE:

- We do not abort if we made a good guess, *i.e,* $id^\star = id_i$. As i is perfectly hidden from $\mathcal{A}$, the probability that we do not abort is $\frac{1}{q_H}$.

– Conditioned on not aborting, the view we provides to $\mathcal{A}$ is exactly the same as in the previous game. We get thus the same advantage in attacking RankPKE as $\mathcal{A}$'s advantage in attacking IBE

We finally have:

$$| \, 2 \Pr[\mathsf{S}_3] - 1 \, | \leq q_H \epsilon_{\mathsf{pke}}$$

6 Parameters

In this section, we explain how to construct a set of parameters and give an analysis of the best known attacks against the IBE scheme.

6.1 General parameters for RankSign and RankEnc

First, we have to carefully choose the parameters of the algorithm RankSign [26] used for the presampling phase. In the case where only RankPKE is used, the constraints are much weaker. Remember that RankSign is a probabilistic signature algorithm and the probability of returning a valid signature depends on the choice of the parameters. These parameters are:

– q, m : the cardinality of the base field and the degree of the extension field.
– n : the length of the hidden LRPC code used to sign.
– t : the number of random columns added to the LRPC to hide it.
– k, d : the dimension of the LRPC code and the weight of the LRPC code.
– r : the weight of the signature.

The conditions these parameters must verify are [26]

$$n = d(n-k), (r-t)(m-r) + (n-k)(rd-m) = 0, r = t + \frac{n-k}{d}$$

Let us explain the choice of our parameters. First we need to fix d for two reasons:

– if we look at the three conditions, they are homogeneous if d is constant. Thus, we can make other set of parameters from one set by multiply all the parameters (except for d) by a constant.
– d is the weight of the $LRPC^+$ code used for the public master key. It is very important to choose d not too small to ensure the security of the public master key.

Once d is fixed, we can easily test all the valid parameters and choose the most interesting ones, whether we need to optimize the security or the key size.

Then we need to choose the parameters of RankPKE. We need a code which can correct wr errors, where w is the weight of the matrix $\boldsymbol{U}$. We use (n', k', t')-simple codes because because they can asymptotically decode up to d_{GV} errors. In all cases, we have chosen $n' = m$ for simplicity, even if this is not a necessary condition.

Let us describe the size of the keys and of the messages, as well as the computation time of our cryptosystem:

- public master key $\boldsymbol{A}$ is a $(k+t) \times (n+t)$ matrix over $\mathbb{F}_{q^m}$: $(k+t)(n-k)m \lceil \log_2 q \rceil$ bits (under systematic form).
- public key $\boldsymbol{p}_{id}$ is an element of $\mathbb{F}_{q^m}^{n+t}$: $(n+t)m \lceil \log_2 q \rceil$ bits.
- secrete key $\boldsymbol{s}_{id}$ is an element of $\mathbb{F}_{q^m}^{n-k}$: $(n-k)m \lceil \log_2 q \rceil$ bits.
- plaintext $\boldsymbol{m}$ is an element of $\mathbb{F}_{q^m}^{k'}$: $k'm \lceil \log_2 q \rceil$ bits.
- ciphertext is a $(k+t+1) \times n'$ matrix over $\mathbb{F}_{q^m}$: $(k+t+1)n'm \lceil \log_2 q \rceil$ bits.
- to generate the secret key, we need to invert a syndrome with RankSign which takes $(n-k)(n+t)$ multiplications in $\mathbb{F}_{q^m}$ ([26]).
- encryption consists in a multiplication of two matrices of respective sizes $(k+t+1) \times (n+t)$ and $(n+t) \times n'$, which takes $(k+t+1)(n+t)n'$ multiplications in $\mathbb{F}_{q^m}$.
- decryption consists in a multiplication matrix-vector and the decoding of an error of weight wr with a (n', k', t')-simple code, which takes $(k+t+1)n'$ multiplications in $\mathbb{F}_{q^m}$ and $\mathcal{O}\big(((n'-t')wr)^3\big)$ operations in $\mathbb{F}_q$.

A multiplication in $\mathbb{F}_{q^m}$ costs $\tilde{\mathcal{O}}\big(m \log q\big)$ operations in $\mathbb{F}_2$ [44].

6.2 Practical evaluation of the security

In order to analyze the security of the IBE, we recall the result of the Theorem 4: $\epsilon_{\mathsf{ibe}} \leq \frac{2q_H}{q} + \epsilon_{\mathsf{lrpc}^+} + q_H(\epsilon_{\mathsf{drsd}} + \epsilon_{\mathsf{pke}})$. We want $\epsilon_{\mathsf{ibe}} \leqslant 2^{-\lambda}$, where λ is the security parameter. Since the first term only depends on q and on the number of queries, we need $q > q_H 2^{\lambda+1}$. We stress that the size of the data and the computation time are linear in the logarithm of q. In consequence, it is not a problem to have q exponential in the security parameter. Moreover, since all combinatorial attacks are polynomial in q, they are utterly inefficient to break the IBE.

The second type of attacks are the algebraic attacks. An adversary can either attack the public masterkey $\boldsymbol{A}$ by solving an instance of LRPC^+ problem, a public key $\boldsymbol{p}$ of an user by solving an instance of DRSD or a ciphertext by solving an instance of RSL. By using the results in [6], we can estimate the complexity of the attacks and adapt the parameters in consequence.

We give an example of a set of parameters in the following table. We take the standard values $\lambda = 128$ for the security parameter and $q_H = 2^{60}$.

n	$n-k$	m	q	d	t	r	d_{GV}	d_{Sing}	Public masterkey Size (Bytes)	n'	k'	t'	w	Probability of failure
100	20	96	2^{192}	5	12	16	11	20	4,239,360	96	9	66	4	2^{-576}

The decoding algorithm for the simple codes is probabilistic, that is why there is a probability p_f that the decoding fails. However, $p_f \approx \frac{1}{q^{t'-wr+1}}$, since we have a very large q in this example, p_f is negligible. These parameters are large but still tractable, for a first code-based IBE scheme in post-quantum cryptography.

Acknowledgements. This work has been supported in part by the French ANR projects ALAMBIC (ANR-16-CE39-0006) and ID-FIX (ANR-16-CE39-0004).

The work of Adrien Hauteville and Jean-Pierre Tillich was also supported in part by the European Commission through the ICT programme under contract H2020- ICT-2014-1 645622 PQCRYPTO. The authors would also like to thank warmly Olivier Blazy for helpful discussions and the reviewers for their insightful remarks (and especially the last reviewer for his remarks and his very detailed review that helped a lot to improve the editorial quality of this paper).

References

1. Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (H)IBE in the standard model. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 553–572. Springer, May 2010.
2. Michael Alekhnovich. More on average case vs approximation complexity. *Computational Complexity*, 20(4):755–786, 2011.
3. Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography with constant input locality. In Alfred Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 92–110. Springer, August 2007.
4. Paulo S.L.M Barreto, Rafael Misoczki, and Marcos A. Jr. Simplicio. One-time signature scheme from syndrome decoding over generic error-correcting codes. *Journal of Systems and Software*, 84(2):198–204, 2011.
5. Elwyn Berlekamp, Robert McEliece, and Henk van Tilborg. On the inherent intractability of certain coding problems. *IEEE Trans. Inform. Theory*, 24(3):384–386, May 1978.
6. Luk Bettale. *Cryptanalyse algébrique : outils et applications.* PhD thesis, Université Pierre et Marie Curie - Paris 6, 2012.
7. Dan Boneh. The decision Diffie-Hellman problem. In *International Algorithmic Number Theory Symposium*, pages 48–63. Springer, 1998.
8. Dan Boneh and Xavier Boyen. Efficient selective-ID secure identity based encryption without random oracles. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 223–238. Springer, May 2004.
9. Dan Boneh and Xavier Boyen. Secure identity based encryption without random oracles. In Matthew Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 443–459. Springer, August 2004.
10. Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 213–229. Springer, August 2001.
11. Xavier Boyen. Lattice mixing and vanishing trapdoors: A framework for fully secure short signatures and more. In Phong Q. Nguyen and David Pointcheval, editors, *PKC 2010*, volume 6056 of *LNCS*, pages 499–517. Springer, May 2010.
12. Ran Canetti, Shai Halevi, and Jonathan Katz. A forward-secure public-key encryption scheme. *Journal of Cryptology*, 20(3):265–294, July 2007.
13. David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 523–552. Springer, May 2010.
14. Clifford Cocks. An identity based encryption scheme based on quadratic residues. In Bahram Honary, editor, *8th IMA International Conference on Cryptography and Coding*, volume 2260 of *LNCS*, pages 360–363. Springer, December 2001.
15. Nicolas Courtois, Matthieu Finiasz, and Nicolas Sendrier. How to achieve a McEliece-based digital signature scheme. In Colin Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 157–174. Springer, December 2001.

16. Whitfield Diffie and Martin Hellman. New directions in cryptography. *IEEE transactions on Information Theory*, 22(6):644–654, 1976.
17. Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In G. R. Blakley and David Chaum, editors, *CRYPTO'84*, volume 196 of *LNCS*, pages 10–18. Springer, August 1984.
18. Jean-Charles Faugère, Mohab Safey El Din, and Pierre-Jean Spaenlehauer. Computing loci of rank defects of linear matrices using gröbner bases and applications to cryptology. In *ISSAC, 2010, Proceedings*, pages 257–264, 2010.
19. Jean-Charles Faugère, Françoise Levy-dit Vehel, , and Ludovic Perret. Cryptanalysis of Minrank. In David Wagner, editor, *Advances in Cryptology - CRYPTO 2008*, volume 5157 of *LNCS*, pages 280–296, 2008.
20. Ernest Mukhamedovich Gabidulin. Theory of codes with maximum rank distance. *Problemy Peredachi Informatsii*, 21(1):3–16, 1985.
21. Ernst M. Gabidulin, A. V. Paramonov, and O. V. Tretjakov. Ideals over a non-commutative ring and their applications to cryptography. In *Advances in Cryptology - EUROCRYPT'91*, number 547 in LNCS, pages 482–489, Brighton, April 1991.
22. Philippe Gaborit. Shorter keys for code based cryptography. In *Proceedings of the 2005 International Workshop on Coding and Cryptography (WCC 2005)*, pages 81–91, Bergen, Norway, March 2005.
23. Philippe Gaborit, Adrien Hauteville, and Jean-Pierre Tillich. Ranksynd a PRNG based on rank metric. In *Post-Quantum Cryptography 2016*, pages 18–28, Fukuoka, Japan, February 2016.
24. Philippe Gaborit, Gaétan Murat, Olivier Ruatta, and Gilles Zémor. Low rank parity check codes and their application to cryptography. In *Proceedings of the Workshop on Coding and Cryptography WCC'2013*, Bergen, Norway, 2013. Available on `www.selmer.uib.no/WCC2013/pdfs/Gaborit.pdf`.
25. Philippe Gaborit, Olivier Ruatta, and Julien Schrek. On the complexity of the rank syndrome decoding problem. *IEEE Trans. Information Theory*, 62(2):1006–1019, 2016.
26. Philippe Gaborit, Olivier Ruatta, Julien Schrek, and Gilles Zémor. Ranksign: An efficient signature algorithm based on the rank metric. In *Post-Quantum Cryptography 2014*, volume 8772 of *LNCS*, pages 88–107. Springer, 2014.
27. Philippe Gaborit and Gilles Zémor. On the hardness of the decoding and the minimum distance problems for rank codes. *IEEE Trans. Information Theory*, 62(12):7245–7252, 2016.
28. Craig Gentry. Practical identity-based encryption without random oracles. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 445–464. Springer, May / June 2006.
29. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008.
30. Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.
31. Adrien Hauteville. Décodage en métrique rang et attaques sur un système de chiffrement à base de codes LRPC. Master's thesis, University of Limoges, September 2014.
32. Adrien Hauteville and Jean-Pierre Tillich. New algorithms for decoding in the rank metric and an attack on the LRPC cryptosystem, 2015. abs/1504.05431.

33. Gregory Kabatianskii, Ernst Krouk, and Ben. J. M. Smeets. A digital signature scheme based on random error-correcting codes. In *IMA Int. Conf.*, volume 1355 of *LNCS*, pages 161–167. Springer, 1997.
34. Gregory Kabatianskii, Ernst Krouk, and Ben. J. M. Smeets. *Error Correcting Coding and Security for Data Networks: Analysis of the Superchannel Concept*. John Wiley & Sons, 2005.
35. Françoise Lévy-dit Vehel and Ludovic Perret. Algebraic decoding of codes in rank metric. In *proceedings of YACC06*, Porquerolles, France, June 2006. available on `http://grim.univ-tln.fr/YACC06/abstracts-yacc06.pdf`.
36. Pierre Loidreau. Asymptotic behaviour of codes in rank metric over finite fields. *Des. Codes Cryptogr.*, 71(1):105–118, 2014.
37. Rafael Misoczki, Jean-Pierre Tillich, Nicolas Sendrier, and Paulo S. L. M. Barreto. MDPC-McEliece: New McEliece variants from moderate density parity-check codes. *IACR Cryptology ePrint Archive, Report2012/409*, 2012, 2012.
38. Harald Niederreiter. Knapsack-type cryptosystems and algebraic coding theory. *Problems of Control and Information Theory*, 15(2):159–166, 1986.
39. Ayoub Otmani and Jean-Pierre Tillich. An efficient attack on all concrete KKS proposals. In *Post-Quantum Cryptography 2011*, volume 7071 of *LNCS*, pages 98–116, 2011.
40. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005.
41. Ryuichi Sakai, Kiyoshi Ohgishi, and Masao Kasahara. Cryptosystems based on pairing. In *SCIS 2000*, Okinawa, Japan, January 2000.
42. Adi Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and David Chaum, editors, *CRYPTO'84*, volume 196 of *LNCS*, pages 47–53. Springer, August 1984.
43. Danilo Silva, Frank R. Kschischang, and Ralf Kötter. Communication over finite-field matrix channels. *IEEE Trans. Information Theory*, 56(3):1296–1305, 2010.
44. Joachim von zur Gathen and Jurgen Gerhard. *Modern computer algebra*. Cambridge University Press, 2003.
45. Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 619–636. Springer, August 2009.
46. Brent R. Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 114–127. Springer, May 2005.

A Discussion

A.1 Toward More Advanced Primitives with Rank Metric

The possibility of designing an Identity-based encryption opens a new way to design more advanced primitives in Rank Metric. This was the case with Lattice based Cryptography: after the work of Gentry, Peikert, and Vaikuntanathan, more generalized primitives have been designed, especially in multi-receiver encryption such as broadcast encryption, attribute-based encryption and functional encryption. Our framework can fit these primitives as well. We remark that, in our IBE encryption, for a public key of an identity $H(id)$, one can sample a lot

of corresponding secret keys $(s, e) = f^{-1}(H(id))$ and it turns out that we can use this for broadcast encryption: considering $H(id)$ as the public key for broadcasting and each legitimate user gets a corresponding secret key (s, e). Thus, the broadcasting mode works in an efficient manner. However, the collusion of the users should be investigated very carefully because the adversary can exploit algebraic structure in rank metric to combine different secret keys. This could be a very interesting direction for further research.

A.2 Toward Algorithmic Considerations of the Hardness in Rank Metric

Post-quantum cryptography is extensively been studied in these days and it would be of great interest to have many alternatives. In this paper, we propose to consider algorithmic problems in rank metric as candidates to be largely studied by the community. The problems in rank metric have been introduced for a long time and their hardness has been proven in some cases, and all known attacks are of exponential complexity. One of the reasons why rank metric has still received too little attention might be due to the fact that it has not been known how to design advanced primitives such as Identity based Encryption with it. In proposing a construction of IBE and providing a rather thorough study of the attacks over the rank metric, we expect more research on the hardness of algorithmic problems in rank metric and the versatile use of rank metric in designing advanced primitives in cryptography.

B Proofs of propositions concerning rank metric

B.1 Proof of proposition 1

Proof. Let $\boldsymbol{e}$ be an error chosen uniformly at random among the elements of $\mathbb{F}_{q^m}^n$ of rank weight w. This means that its coordinates generate over $\mathbb{F}_q$ a subspace E of dimension w.

Let $\boldsymbol{s} = \boldsymbol{H}\boldsymbol{e}$ be the associated error syndrome. Because of the particular form of $\boldsymbol{H}$ we know that the first t coordinates of $\boldsymbol{e}$, i.e. $e_1, \ldots, e_t$ are equal to the t first coordinates $s_1, \ldots, s_t$ of $\boldsymbol{s}$. This gives $e_1, \ldots, e_t$ directly. With probability $1 - \frac{1}{q^{t-w+1}} + o\left(\frac{1}{q^{t-w+1}}\right)$ these coordinates generate E, the support of the error. In such a case, the remaining e_i's can be written as $e_i = \sum_{j=1}^{w} e_{ij} E_j$ with $(E_1, \ldots, E_w)$ a basis of E and $e_{ij} \in \mathbb{F}_q$.

The first t coordinates of the syndrome $\boldsymbol{s}$ have been used to recover E, now the last $n - k - t$ coordinates can be used to recover the error coordinates e_{ij}. This is done by solving the set of $(n - k - t)m$ linear equations in the $(n - t)w$ unknowns (the e_{ij} for $i > t$) obtained by writing the last $n - k - t$ syndrome equations over $\mathbb{F}_q$.

Now, the value $\frac{m+n-\sqrt{(m-n)^2+4km}}{2}$ is the smallest root of the polynomial $P(x) =$

$(n-k-x)m-(n-x)x$.

$$w \leqslant t < \frac{m+n-\sqrt{(m-n)^2+4km}}{2} \Rightarrow (n-k-t)m > (n-t)t \geqslant (n-w)t.$$

Thus, the system has more equations than unknowns. By construction it always has a solution. This solution is unique if and only if the system is full-rank, which happens with probability $\approx 1-\frac{1}{q^{(n-k-t)m-(n-t)w+1}} > 1-\frac{1}{q^{t-w+1}}$. □

B.2 Proof of the hardness of DRSD problem 5

Proof. First we need a way to embed an $\mathbb{F}_q$-linear code into an $\mathbb{F}_{q^m}$-linear code. We use the same technique as in [27].

Definition 12. *Let $m \geqslant n$ and $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{F}_{q^m}^n$. We define the embedding of $\mathbb{F}_q^n$ into $\mathbb{F}_{q^m}^n$ by :*

$$\begin{aligned} \psi_\alpha : \quad \mathbb{F}_q^n &\to \mathbb{F}_{q^m}^n \\ (x_1, \ldots, x_n) &\mapsto (\alpha_1 x_1, \ldots, \alpha_n x_n) \end{aligned} \tag{4}$$

For every $\mathbb{F}_q$-linear code $\mathcal{C}$, we denote by $C(\mathcal{C}, \alpha)$ the $\mathbb{F}_{q^m}$-linear code generated by the set $\psi_\alpha(\mathcal{C})$.

According to Theorem 8 of [27], if we take $m > 4qn$, the probability that the minimum rank distance of $C(\mathcal{C}, \alpha)$ differs from the minimum Hamming distance of $\mathcal{C}$ decreases exponentially with n. Now let $(\boldsymbol{G}, \boldsymbol{y})$ be an instance of DRSD and $\alpha \in \mathbb{F}_{2^m}^n$ such that its coordinates are independent over $\mathbb{F}_q$. There are two possible cases :

- $\boldsymbol{y} = \boldsymbol{m}\boldsymbol{G} + \boldsymbol{x}$ with $|\boldsymbol{x}| = r$ (where $|\boldsymbol{x}|$ stands for the Hamming weight of $\boldsymbol{x}$), then $\psi_\alpha(\boldsymbol{y}) = \boldsymbol{m}\psi_\alpha(\boldsymbol{G}) + \psi_\alpha(\boldsymbol{x})$. Since $\|\psi_\alpha(\boldsymbol{x})\| = r$ and the minimum rank distance of the code generated by $\psi_\alpha(\boldsymbol{G})$ is equal to the minimum Hamming distance of the code generated by $\boldsymbol{G}$, D_R accepts the input $(\psi_\alpha(\boldsymbol{G}), \psi_\alpha(\boldsymbol{y}))$ with a non negligible advantage.
- $\boldsymbol{y}$ is random then $\psi_\alpha(\boldsymbol{y})$ is also random and D_R accepts $(\psi_\alpha(\boldsymbol{G}), \psi_\alpha(\boldsymbol{y}))$ with probability $1/2$.

Thus, the DSD problem is probabilistically reduced to the DRSD problem, which proves that this problem is hard. □

C Proof of Theorem 2 and a related result

Proof of lemma 1

Proof. $\mathcal{C}$ is clearly an $\mathbb{F}_q$-linear code, since it is defined by linear equations over $\mathbb{F}_q$. W_U is an $\mathbb{F}_q$-linear space of dimension at most N and if we let $\mathcal{C}_0$ be the $\mathbb{F}_{q^m}$-linear code of dimension k over $\mathbb{F}_{q^m}$ and km over $\mathbb{F}_q$ defined by

$$\mathcal{C}_0 \stackrel{\text{def}}{=} \{\boldsymbol{x} \in \mathbb{F}_{q^m}^n : \boldsymbol{A}\boldsymbol{x} = 0\}$$

we have $\dim_{\mathbb{F}_q} \mathcal{C} \leq \dim_{\mathbb{F}_q} \mathcal{C}_0 + \dim_{\mathbb{F}_q} W_U \leq km + N$. Obviously we have that $\mathcal{C}'$ is contained in $\mathcal{C}$ and the vector $\sum_i \alpha_i \boldsymbol{u}_i$ has all its entries in V when the α_i's belong to $\mathbb{F}_q$. Therefore it is of rank weight at most $\dim V = w$. □

C.1 A first crude analysis of Algorithm 1

Before giving the proof of Theorem 2 it might be insightful to give a first hint explaining why Algorithm 1 behaves better when applied to $\mathcal{C}$ than for generic codes of the same length and dimension as C but with just a single codeword of rank weight w (up to multiplication by elements of $\mathbb{F}_q$). This is provided by the following lemma.

Lemma 2. *Assume that the $\boldsymbol{u}_i$'s are independent over $\mathbb{F}_q$. $\mathcal{C}'$ always contains an non-zero element of rank $\leq w - \delta$ where $\delta = \lfloor \frac{N}{n} \rfloor$.*

Proof. Consider an $\mathbb{F}_q$ linear mapping φ from $\mathbb{F}_{q^m}$ to $\mathbb{F}_q^w$ which is such that its restriction to V is one-to-one and onto. This amounts to $\varphi(V) = \mathbb{F}_q^w$. By some abuse of notation we also denote by φ its natural extension to $\mathbb{F}_{q^m}^n$. This is a mapping from $\mathbb{F}_{q^m}^n$ to $\mathbb{F}_q^{w\times n}$ where $\varphi(v_1, \ldots, v_n)$ is defined as the matrix with columns equal to $\varphi(v_i)$ (and by viewing $\varphi(v_i)$ as a column vector). Consider now the matrix code $\varphi(\mathcal{C}') = \{\varphi(\boldsymbol{c}) : \boldsymbol{c} \in \mathcal{C}'\} \subset \mathbb{F}_q^{w\times n}$. Notice that $\varphi(\boldsymbol{c})$ has the same rank as $\boldsymbol{c}$ since the dimension of the column space of $\phi(\boldsymbol{c})$ is the same as the dimension as the vector space (over $\mathbb{F}_q$) generated by the entries c_i of $\boldsymbol{c}$ since φ is an isomorphism from V to $\mathbb{F}_q^w$. This implies that the rank distance of $\mathcal{C}'$ and $\varphi(\mathcal{C}')$ are the same. The assumption on the independence of the $\boldsymbol{u}_i$'s implies that the dimension of $\mathcal{C}'$ over $\mathbb{F}_q$ is N. The Singleton bound on the rank distance of a code of dimension N over $\mathbb{F}_q$ implies that the minimum rank distance of $\varphi(\mathcal{C}')$ is $\leq w - \delta$. This implies that the rank distance of $\varphi(\mathcal{C}')$ is also less than or equal to $w - \delta$ and that $\mathcal{C}'$ contains a non zero codeword of rank $\leq w - \delta$.

In other words when we look for elements of $\mathcal{C}$ which might lie in $\mathcal{C}'$ we can look for codewords of rank $w - \delta$. Algorithm 1 is nothing but Algorithm 1 of [32] (which is just the adaptation of the support attack of [25, Sec. IV.] to codes which are $\mathbb{F}_q$-linear). This algorithm is analyzed there and we expect to find such codewords with complexity $\mathcal{O}\big((n-k)m+N)^3 q^{(w-\delta)\lceil \frac{N+km}{n} \rceil}\big)$ when $n \geq m$. When N and km are multiple of n, we obtain (by ignoring the polynomial multiplicative constant) and by setting $\alpha \stackrel{\text{def}}{=} \frac{m}{n}$, $R \stackrel{\text{def}}{=} \frac{k}{n}$ a complexity of order $\tilde{\mathcal{O}}\big(q^{(w-\delta)(\delta+\alpha Rn)}\big)$. This should be compared to the complexity of obtaining a codeword of weight w in an $\mathbb{F}_q$ linear code which is $\tilde{\mathcal{O}}\big(q^{w\alpha Rn}\big)$ and $\tilde{\mathcal{O}}\big(q^{(w-1)\alpha Rn}\big)$ for an $\mathbb{F}_{q^m}$-linear code (see [32, prop. 4.1]). When $1 \ll \delta \ll n$ the (additive) gain in the exponent is roughly $\delta\alpha Rn = \alpha RN$. In other words, the complexity gets reduced by a multiplicative factor of order $q^{\alpha RN}$. This is less than q^N which might very well be the best we could hope for.

This is a very crude analysis which shows in a simple way why we gain something in the case at hand, there are namely so many codewords in $\mathcal{C}$ whose entries all lie in this subspace V (i.e. the subcode $\mathcal{C}'$) that there should be codewords of weight strictly less than w and which are therefore easier to find than codewords of rank weight w. Actually that Theorem 2 does is to analyze Algorithm 1 properly and this will improve a little bit the aforementioned complexity.

C.2 Proof of Theorem 2

When we apply Algorithm 1 on the aforementioned code $\mathcal{C}$, it works even better than what is predicted by Lemma 2. It is straightforward to check the following fact.

Lemma 3. *In the list of candidates that Algorithm 1 can output when the linear system (3) has a solution, there is an element of $\mathcal{C}'$ if and only if there is an non-zero element $\boldsymbol{c}' = (c'_i)_{1 \leq i \leq n}$ in $\mathcal{C}'$ such that the subspace of $\mathbb{F}_{q^m}$ generated (over $\mathbb{F}_q$) by the entries c'_i is contained in $W \cap V$.*

Proof. Assume that a non-zero element $\boldsymbol{c}' = (c'_1, \ldots, c_n)$ of $\mathcal{C}'$ can be output by Algorithm 1. Then necessarily we have that the subspace (over $\mathbb{F}_q$) generated by its entries should belong to W. Since these entries belong to V by assumption, these entries also belong to $W \cap V$. The converse follows from the same reasons.

This raises the issue to quantify the probability of such an event. This depends on the dimension of $W \cap V$ as explained by the following lemma.

Lemma 4. *Let $w - \Delta$ be the dimension of $W \cap V$. The probability p (taken over the random choices of the errors forming U) that there exists an non-zero element $\boldsymbol{c}' = (c'_i)_{1 \leq i \leq n}$ in $\mathcal{C}'$ such that the subspace of $\mathbb{F}_{q^m}$ generated (over $\mathbb{F}_q$) by the entries c'_i is contained in $W \cap V$ satisfies*

$$p = \Theta\left(q^{\min(N - n\Delta, 0)}\right).$$

Proof. Let $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_N$ be the N errors that have been chosen for forming the matrix $\boldsymbol{U}$. Since $W \cap V$ is of dimension Δ there exists an $\mathbb{F}_q$-linear map L from V to $\mathbb{F}_q^{\Delta}$ whose kernel is precisely $W \cap V$. By some abuse of notation, we also denote by $f(\boldsymbol{u}_i)$ the vector in $\mathbb{F}_q^{\Delta n}$ formed by concatenating the entries of $f(u_{i1}), f(u_{i2}), \ldots, f(u_{in})$. Let M be the $n\Delta \times N$ matrix with columns $f(\boldsymbol{u}_1), \ldots, f(\boldsymbol{u}_N)$. From the way the $\boldsymbol{u}_i$'s are chosen, it turns out M is drawn uniformly at random among the set of matrices of size $n\Delta \times N$ over $\mathbb{F}_q$. There exists an non-zero element $\boldsymbol{c}' = (c'_i)_{1 \leq i \leq n}$ in $\mathcal{C}'$ such that the subspace of $\mathbb{F}_{q^m}$ generated (over $\mathbb{F}_q$) by the entries c'_i is contained in $W \cap V$ if and only if the columns of this matrix are not independent. The probability of this event is 1 if $N > \delta n$ and is $\Theta(q^{N - \delta n})$ otherwise.

The last lemma we will need is to quantify the probability that the subspaces W and V have an intersection of a certain dimension. This result is probably well known, we have found a trace of this result together with a proof in [31].

Lemma 5. *For any $\Delta \in \{0, 1, \ldots, \min(m - r, w)\}$ we have*

$$\mathrm{Prob}(\dim W \cap V = w - \Delta) = \Theta\left(q^{(w-\Delta)(r-m+\Delta)}\right).$$

We are ready now analyzing the complexity of Algorithm 1 when applied to $\mathcal{C}$.

Theorem 5. *Assume that* $w \leq \min\left(\left\lfloor \frac{K}{n} \right\rfloor, \left\lfloor \frac{N}{n} \right\rfloor + 1\right)$ *and that* $\frac{w+\left\lfloor \frac{K}{n} \right\rfloor}{2} \geq \left\lfloor \frac{N}{n} \right\rfloor$. *Let*

$$e_- = \left(w - \left\lfloor \frac{N}{n} \right\rfloor\right)\left(\left\lfloor \frac{K}{n} \right\rfloor - \left\lfloor \frac{N}{n} \right\rfloor\right)$$
$$e_+ = \left(w - \left\lfloor \frac{N}{n} \right\rfloor - 1\right)\left(\left\lfloor \frac{K}{n} \right\rfloor - \left\lfloor \frac{N}{n} \right\rfloor - 1\right) + n\left(\left\lfloor \frac{N}{n} \right\rfloor + 1\right) - N$$

Algorithm 1 outputs an element of $\mathcal{C}'$ *with complexity* $\tilde{\mathcal{O}}\left(q^{\min(e_-,e_+)}\right)$.

Remark 3.

1. When N and $K = km + N$ are multiple of n, say $N = \delta n$ and $K = \alpha R n + \delta$ (with $\alpha \stackrel{\text{def}}{=} \frac{m}{n}$, $R = \frac{k}{n}$) the complexity above simplifies to $\tilde{\mathcal{O}}\left(q^{\alpha R n(w-\delta)}\right)$ which is slightly better than the bound on the complexity following from Lemma 2 which is of the form $\tilde{\mathcal{O}}\left(q^{(\delta+\alpha R n)(w-\delta)}\right)$.
2. This approach is really suited to the case $m \leq n$. When $m > n$ we obtain better complexities by working on the transposed code (see [32] for more details about this approach).

Proof. Let us first compute the probability p that there is an non-zero element $\boldsymbol{c}' = (c'_i)_{1 \leq i \leq n}$ in $\mathcal{C}'$ such that the subspace of $\mathbb{F}_{q^m}$ generated (over $\mathbb{F}_q$) by the entries c'_i is contained in $W \cap V$. Let $p(\Delta)$ be this probability when the dimension of $W \cap V$ is equal to $w - \Delta$. We have

$$\begin{aligned}
p &= \sum_{\Delta=0}^{w} p(\Delta) \operatorname{Prob}(\dim W \cap V = w - \Delta) \\
&= \Theta\left(q^{\min(N-n\Delta,0)}\right)\Theta\left(q^{(w-\Delta)(r-m+\Delta)}\right) \text{ (by Lemmas 4 and 5)} \\
&= \Theta\left(q^{(w-\Delta)(r-m+\Delta)+\min(N-n\Delta,0)}\right) \\
&= \Theta\left(q^{(w-\Delta)\left(-\left\lfloor \frac{K}{n} \right\rfloor+\Delta\right)+\min(N-n\Delta,0)}\right) \text{ (since } m - r = \left\lfloor \frac{K}{n} \right\rfloor\text{)} \\
&= \Theta\left(q^{f(\Delta)}\right),
\end{aligned}$$

where f is the function defined over the positive integers by

$$\begin{aligned}
f(x) &= (w-x)\left(-\left\lfloor \frac{K}{n} \right\rfloor + x\right) \text{ for } x \in \left\{0, \ldots, \left\lfloor \frac{N}{n} \right\rfloor\right\} \\
&= (w-x)\left(-\left\lfloor \frac{K}{n} \right\rfloor + x\right) + N - nx \text{ for } x \geq \left\lfloor \frac{N}{n} \right\rfloor + 1.
\end{aligned}$$

Let g and h be defined by

$$g(x) = (w-x)\left(-\left\lfloor \frac{K}{n} \right\rfloor + x\right)$$
$$h(x) = (w-x)\left(-\left\lfloor \frac{K}{n} \right\rfloor + x\right) + N - nx.$$

Note that

$$g(x) = -x^2 + \left(w + \left\lfloor \frac{K}{n} \right\rfloor\right) x - w \left\lfloor \frac{K}{n} \right\rfloor$$

It is easy to see that the maximum of the first expression is attained at $x = \left\lfloor \frac{N}{n} \right\rfloor$ since g is strictly increasing on $[0, \left\lfloor \frac{N}{n} \right\rfloor]$ (it attains its minimum at $x_0 \stackrel{\text{def}}{=} \frac{w + \left\lfloor \frac{K}{n} \right\rfloor}{2}$ which is larger than $\left\lfloor \frac{N}{n} \right\rfloor$ by hypothesis). It is equal to $-e_-$. Similarly the maximum of the second expression is attained at $x = \left\lfloor \frac{N}{n} \right\rfloor + 1$ and is equal to $-e_+$. From this we deduce that

$$p = \Theta\left(q^{\max(-e_-, -e_+)}\right). \tag{5}$$

With constant and non zero probability the dimension of the solution space of (3) is equal to 1 when it is not reduced to the zero vector. In such a case and when there is a non-zero element $\boldsymbol{c}' = (c'_i)_{1 \le i \le n}$ in $\mathcal{C}'$ such that the subspace of $\mathbb{F}_{q^m}$ generated (over $\mathbb{F}_q$) by the entries of c'_i is contained in $W \cap V$, the algorithm outputs an element of $\mathcal{C}'$. The complexity of our algorithm is therefore given by $\mathcal{O}\left(\frac{1}{p}\right)$ (by ignoring polynomial terms consisting in solving the linear system (3)) which yields the aforementioned result.