Rank Quasi-Cyclic (RQC)

RQC is an IND-CCA2 KEM running for standardization to NIST's competition in the category "post-quantum public key encryption scheme". Different sets of parameters are proposed for security strength categories 1, 3, and 5.

Principal Submitters (by alphabetical order):

- Carlos AGUILAR MELCHOR
- Nicolas ARAGON
- Slim BETTAIEB
- Loïc BIDOUX
- Olivier BLAZY
- Jean-Christophe DENEUVILLE
- Philippe GABORIT
- Gilles ZÉMOR

Inventors: Same as submitters

Developers: Same as submitters

Owners: Same as submitters

Main contact

🚹 Philippe GABORIT
@ philippe.gaborit@unilim.fr
☎ +33-626-907-245
≘ University of Limoges
✉ 123 avenue Albert Thomas
87 060 Limoges Cedex
France

Backup point of contact

🚹 Jean-Christophe DENEUVILLE
@ jch.deneuville@gmail.com
☎ +33-631-142-705
≘ INSA Centre Val de Loire
✉ 4 rue Jean le Bail
87 000 Limoges
France

Signatures

Digital copies of the signed statements are provided in Appendix A. The original paper versions will be given to Dustin MOODY directly at the First PQC Standardization Conference.

Contents

1 Specifications

In this section, we introduce RQC, an efficient encryption scheme based on coding theory. RQC stands for Rank Quasi-Cyclic. This proposal is currently under revision for publication in IEEE Transactions on Information Theory. Many notations, definitions and properties are very similar to [6]. We nevertheless include them in this proposal for completeness.

RQC is a code-based public key cryptosystem with several desirable properties:

- It is proved IND-CPA assuming the hardness of (a decisional version of) the Syndrome Decoding on structured codes. By construction, RQC perfectly fits the recent KEM-DEM transformation of [17], and allows to get an hybrid encryption scheme with strong security guarantees (IND-CCA2) and good efficiency,
- In contrast with most code-based cryptosystems, the assumption that the family of codes being used is indistinguishable among random codes is no longer required, and
- It features more attractive parameters than most of the Hamming based proposals.

Organization of the Specifications. This section is organized as follows: we provide the required background in Sec. 1.1, we make some recalls on encryption and security in Sec. 1.1.4 then present our proposal in Sec. 1.2. Concrete sets of parameters are provided in Sec. 1.3.

1.1 Preliminaries

1.1.1 General definitions

Throughout this document, $\mathbb{Z}$ denotes the ring of integers and for $m, q \in \mathbb{Z}$, q prime, $\mathbb{F}_{q^m}$ denotes an extension of degree m of the finite field of q elements. Additionally, we denote by $\omega(\cdot)$ the rank weight of a vector (see Def. 1.1.10), and by $\mathcal{S}_w^n\left(\mathbb{F}_{q^m}\right)$ the set of words in $\mathbb{F}_{q^m}^n$ of weight w. Formally:

$$\mathcal{S}_w^n\left(\mathbb{F}_{q^m}\right) = \left\{\mathbf{v} \in \mathbb{F}_{q^m}^n, \text{ such that } \omega(\mathbf{v}) = w\right\}.$$

Let $\mathcal{V}$ denotes a vector space of dimension n over some finite field $\mathbb{F}$ for some positive $n \in \mathbb{Z}$. Elements of $\mathcal{V}$ can be interchangeably considered as row vectors or polynomials in $\mathcal{R} = \mathbb{F}[X]/(X^n - 1)$. Vectors/Polynomials (resp. matrices) will be represented by lower-case (resp. upper-case) bold letters. A prime integer n is said primitive if the polynomial $(X^n - 1)/(X - 1)$ is irreducible in $\mathcal{R}$.

For $\mathbf{u}, \mathbf{v} \in \mathcal{V}$, we define their product similarly as in $\mathcal{R}$, *i.e.* $\mathbf{uv} = \mathbf{w} \in \mathcal{V}$ with

$$w_k = \sum_{i+j \equiv k \mod n} x_i y_j, \text{ for } k \in \{0, 1, \ldots, n-1\}. \tag{1}$$

Our new protocol takes great advantage of the cyclic structure of matrices. In the same fashion as [1], $\mathbf{rot}(\mathbf{h})$ for $\mathbf{h} \in \mathcal{V}$ denotes the circulant matrix whose i^{th} column is the vector corresponding to $\mathbf{h}X^i$. This is captured by the following definition.

Definition 1.1.1 (Circulant Matrix)**.** *Let* $\mathbf{v} = (v_1, \dots, v_n) \in \mathbb{F}_q^n$*. The* circulant matrix *induced by* $\mathbf{v}$ *is defined and denoted as follows:*

$$\mathbf{rot}(\mathbf{v}) = \begin{pmatrix} v_0 & v_{n-1} & \dots & v_1 \\ v_1 & v_0 & \dots & v_2 \\ \vdots & \vdots & \ddots & \vdots \\ v_{n-1} & v_{n-2} & \dots & v_0 \end{pmatrix} \in \mathbb{F}_{||}^{n \times n} \tag{2}$$

As a consequence, it is easy to see that the product of any two elements $\mathbf{u}, \mathbf{v} \in \mathcal{R}$ can be expressed as a usual vector-matrix (or matrix-vector) product using the $\mathbf{rot}(\cdot)$ operator as

$$\mathbf{u} \cdot \mathbf{v} = \mathbf{u} \times \mathbf{rot}(\mathbf{v})^\top = \left(\mathbf{rot}(\mathbf{u}) \times \mathbf{v}^\top\right)^\top = \mathbf{v} \times \mathbf{rot}(\mathbf{u})^\top = \mathbf{v} \cdot \mathbf{u}. \tag{3}$$

Coding Theory. We now recall some basic definitions and properties about coding theory that will be useful to our construction. We mainly focus on general definitions, and refer the reader to Sec. 1.2 for the description of the scheme, and also to [18] for a complete survey on code-based cryptography.

Definition 1.1.2 (Linear Code)**.** *A* Linear Code $\mathcal{C}$ *of length* n *and dimension* k *(denoted* $[n, k]$*) is a subspace of* $\mathcal{R}$ *of dimension* k*. Elements of* $\mathcal{C}$ *are referred to as codewords.*

Definition 1.1.3 ($\mathbb{F}_{q^m}$-linear code)**.** *An* $\mathbb{F}_{q^m}$*-linear code* $\mathcal{C}$ *of length* n *and dimension* k *is a linear subspace of* $\mathbb{F}_{q^m}^n$ *of dimension* k*. We denote it* $\mathcal{C}[n, k]_{q^m}$ *or simply* $\mathcal{C}[n, k]$ *if the context is clear.*

Definition 1.1.4 (Generator Matrix)**.** *We say that* $\mathbf{G} \in \mathbb{F}^{k \times n}$ *is a* Generator Matrix *for the* $[n, k]$ *code* $\mathcal{C}$ *if*

$$\mathcal{C} = \left\{\mathbf{mG}, \text{ for } \mathbf{m} \in \mathbb{F}^k\right\}. \tag{4}$$

Definition 1.1.5 (Parity-Check Matrix)**.** *Given an* $[n, k]$ *code* $\mathcal{C}$*, we say that* $\mathbf{H} \in \mathbb{F}^{(n-k) \times n}$ *is a* Parity-Check Matrix *for* $\mathcal{C}$ *if* $\mathbf{H}$ *is a generator matrix of the dual code* $\mathcal{C}^\perp$*, or more formally, if*

$$\mathcal{C}^\perp = \left\{\mathbf{v} \in \mathbb{F}^n \text{ such that } \mathbf{Hv}^\top = \mathbf{0}\right\}, \tag{5}$$

where $\mathbf{Hv}^\top$ *is the* syndrome *of* $\mathbf{v}$*.*

Definition 1.1.6 (Minimum Distance)**.** *Let* $\mathcal{C}$ *be an* $[n, k]$ *linear code over* $\mathcal{R}$ *and let* ω *be a norm on* $\mathcal{R}$ *(the rank weight for us, see Def. 1.1.10). The* Minimum Distance *of* $\mathcal{C}$ *is*

$$d = \min_{\mathbf{u}, \mathbf{v} \in \mathcal{C}, \mathbf{u} \neq \mathbf{v}} \omega(\mathbf{u} - \mathbf{v}). \tag{6}$$

A code with minimum distance d is capable of decoding arbitrary patterns of up to $\delta = \lfloor \frac{d-1}{2} \rfloor$ errors. Code parameters are denoted $[n, k, d]$.

Code-based cryptography usually suffers from huge keys. In order to keep our cryptosystem efficient, we will use the strategy of Gaborit [9] for shortening keys. This results in Quasi-Cyclic Codes, as defined below.

Definition 1.1.7 (Quasi-Cyclic Codes [22])**.** *View a vector* $\mathbf{c} = (\mathbf{c}_1, \dots, \mathbf{c}_s)$ *of* $\mathbb{F}_2^{sn}$ *as* s *successive blocks (n-tuples). An* $[sn, k, d]$ *linear code* $\mathcal{C}$ *is* Quasi-Cyclic (QC) *of index* s *if, for any* $\mathbf{c} = (\mathbf{c}_1, \dots, \mathbf{c}_s) \in \mathcal{C}$*, the vector obtained after applying a simultaneous circular shift to every block* $\mathbf{c}_1, \dots, \mathbf{c}_s$ *is also a codeword.*

More formally, by considering each block $\mathbf{c}_i$ *as a polynomial in* $\mathcal{R} = \mathbb{F}[X]/(X^n - 1)$*, the code* $\mathcal{C}$ *is QC of index* s *if for any* $\mathbf{c} = (\mathbf{c}_1, \dots, \mathbf{c}_s) \in \mathcal{C}$ *it holds that* $(X \cdot \mathbf{c}_1, \dots, X \cdot \mathbf{c}_s) \in \mathcal{C}$*.*

Definition 1.1.8 (Systematic Quasi-Cyclic Codes)**.** *A* systematic *Quasi-Cyclic* $[sn, n]$ *code of index* s *and rate* $1/s$ *is a quasi-cyclic code with an* $(s-1)n \times sn$ *parity-check matrix of the form:*

$$\mathbf{H} = \begin{bmatrix} \mathbf{I}_n & 0 & \cdots & 0 & \mathbf{A}_1 \\ 0 & \mathbf{I}_n & & & \mathbf{A}_2 \\ & & \ddots & & \vdots \\ 0 & & \cdots & \mathbf{I}_n & \mathbf{A}_{s-1} \end{bmatrix} \tag{7}$$

where $\mathbf{A}_1, \dots, \mathbf{A}_{s-1}$ *are circulant* $n \times n$ *matrices.*

Remark 1.1. *The definition of systematic quasi-cyclic codes of index* s *can of course be generalized to all rates* ℓ/s*,* $\ell = 1 \dots s-1$*, but we shall only use systematic QC-codes of rates* $1/2$ *and* $1/3$ *and wish to lighten notation with the above definition. In the sequel, referring to a systematic QC-code will imply by default that it is of rate* $1/s$*. Note that arbitrary QC-codes are not necessarily equivalent to a systematic QC-code.*

The definitions usually associated to Hamming metric codes such as norm (Hamming weight), support (non-zero coordinates), and isometries ($n \times n$ permutation matrices) can be adapted to the rank metric setting based on the representation of elements as matrices in $\mathbb{F}_q^{m \times n}$.

We recall some definitions and properties of rank metric codes, and refer the reader to [20] for more details. Consider the case where $\mathbb{F}$ is an extension of a finite field, *i.e.* $\mathbb{F} = \mathbb{F}_{q^m}$, and let $\mathbf{x} = (x_1, \dots, x_n) \in \mathbb{F}_{q^m}^n$ be an element of some vector space of dimension n over $\mathbb{F}_{q^m}$. A basic property of field extensions is that they can be seen as vector spaces over the base field they extend. Hence, by considering $\mathbb{F}_{q^m}$ as a vector space of dimension m over $\mathbb{F}_q$, and given a basis $(\mathbf{e_1}, \dots, \mathbf{e_m}) \in \mathbb{F}_q^m$, one can express each x_i as

$$x_i = \sum_{j=1}^{m} x_{j,i} \mathbf{e_j} \text{ (or equivalently } x_i = (x_{1,i}, \dots, x_{m,i})\text{)}. \tag{8}$$

Using such an expression, we can expand $\mathbf{x} \in \mathbb{F}_{q^m}^n$ to a matrix $\mathbf{E}(\mathbf{x})$ such that:

$$\mathbf{x} = \begin{pmatrix} x_1 & x_2 & \dots & x_n \end{pmatrix} \in \mathbb{F}_{q^m}^n \tag{9}$$

$$\mathbf{E}(\mathbf{x}) = \begin{pmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,n} \\ x_{2,1} & x_{2,2} & \dots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m,1} & x_{m,2} & \dots & x_{m,n} \end{pmatrix} \in \mathbb{F}_q^{m \times n}. \tag{10}$$

For an element $\mathbf{x}$ of $\mathbb{F}_{q^m}^n$ we define its rank norm $\omega(\mathbf{x})$ as the rank of the matrix $\mathbf{E}(\mathbf{x})$. A rank metric 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 norm. In the following, $\mathcal{C}$ is a rank metric code of length n and dimension k over $\mathbb{F}_{q^m}$, where $q = p^\eta$ for some prime p and positive $\eta \geq 1$. The matrix $\mathbf{G}$ denotes a $k \times n$ generator matrix of $\mathcal{C}$.

The minimum rank distance of the code $\mathcal{C}$ is the minimum rank of non-zero vectors of the code. We also considers the usual inner product which allows to define the notion of dual code.

Let $\mathbf{x} = (x_1, x_2, \cdots, x_n) \in \mathbb{F}_{q^m}^n$ be a vector of rank r. We denote by $E = \langle x_1, \ldots, x_n \rangle$ the $\mathbb{F}_q$-subspace of $\mathbb{F}_{q^m}$ generated by the coordinates of $\mathbf{x}$ *i.e.* $E = \text{Vect}\,(x_1, \ldots, x_n)$. The vector space E is called the *support* of $\mathbf{x}$ and denoted $\text{Supp}(\mathbf{x})$. Finally, the notion of *isometry* which in Hamming metric corresponds to the action of the code on $n \times n$ permutation matrices, is replaced for the rank metric by the action of $n \times n$ invertible matrices over the base field $\mathbb{F}_q$.

Definition 1.1.9. *Let* $\mathbf{x} \in \mathbb{F}_{q^m}^n$*. The support of* $\mathbf{x}$ *denoted* $\text{Supp}(\mathbf{x})$*, is the* $\mathbb{F}_q$*-linear space of* $\mathbb{F}_{q^m}$ *spanned by the coordinates of* $\mathbf{x}$*. Formally,*

$$\text{Supp}(\mathbf{x}) = \mathit{Vect}\,(\mathbf{E}\,(\mathbf{x})) = \langle x_1, \ldots, x_n \rangle_{\mathbb{F}_q}$$

The number of supports of dimension w is the number of linear subspaces of $\mathbb{F}_{q^m}$ of dimension w: $\begin{bmatrix} m \\ w \end{bmatrix}_q = \prod_{i=0}^{w-1} \frac{q^m - q^i}{q^w - q^i} = \Theta\big(q^{w(m-w)}\big)$.

Definition 1.1.10 (Rank weight)**.** *The rank weight of a vector* $\mathbf{x} \in \mathbb{F}_{q^m}^n$ *is given by the rank of its matrix* $\boldsymbol{E}(\mathbf{x})$ *as defined in Eq. (10). Therefore,* $\omega(\mathbf{x}) = \mathsf{rank}(\mathbf{E}(\mathbf{x})) = \dim\left(\text{Supp}(\mathbf{x})\right)$*.*

Bounds for rank metric codes. The classical bounds for Hamming metric have straightforward rank metric analogues.

Singleton Bound. The classical Singleton bound for linear $[n, k]$ codes of minimum rank r over $\mathbb{F}_{q^m}$ applies naturally in the rank metric setting. It works in the same way as for linear codes (by finding an information set) and reads $r \leq 1 + n - k$. When $n > m$ this bound can be rewritten [20] as

$$r \leq 1 + \left\lfloor \frac{(n-k)m}{n} \right\rfloor. \tag{11}$$

Codes achieving this bound are called Maximum Rank Distance codes (MRD).

Deterministic Decoding. Unlike the situation for the Hamming metric, there do not exist many families of codes for the rank metric which are able to decode rank errors efficiently up to a given norm. When we are dealing with deterministic decoding, there is essentially only one known family of rank codes which can decode efficiently: the family of Gabidulin codes [7]. More details about these codes are provided in the next subsection.

In a nutshell, they are defined over $\mathbb{F}_{q^m}$ and for $k \leq n \leq m$, Gabidulin codes of length n and dimension k are optimal and satisfy the Singleton bound for $m = n$ with minimum distance $d = n - k + 1$. They can decode up to $\lfloor \frac{n-k}{2} \rfloor$ rank errors in a deterministic way.

Probabilistic Decoding. There also exists a simple family of codes which has been described for the subspace metric in [27] and can be straightforwardly adapted to the rank metric. These codes reach asymptotically the equivalent of the Gilbert-Varshamov bound for the rank metric, however their non-zero probability of decoding failure makes them less interesting for the cases we consider in this paper.

1.1.2 Gabidulin codes and their decoding

Gabidulin codes were introduced in 1985 [7]. These codes are analogs to Reed-Solomon codes in Hamming metric [25], but involves q-polynomials instead of regular ones, and have therefore a strong algebraic structure. q-polynomials were introduced by Ore [23], we hereafter give some background.

Definition 1.1.11 (q-polynomials)**.** *The set of q-polynomials over $\mathbb{F}_{q^m}$ is the set of polynomials with the following shape:*

$$\left\{ P(X) = \sum_{i \in \mathbb{N}} p_i X^{q^i}, \text{ with } (p_i) \in \mathbb{F}_{q^m}^{\mathbb{N}} \text{ of finite support} \right\}$$

The q-degree of a q-polynomial P, denoted $\deg_q(P)$, is the biggest integer r such that $p_r \neq 0$.

Definition 1.1.12 (Gabidulin codes)**.** *Let $k, n, m \in \mathbb{N}$ such that $k \leqslant n \leqslant m$. Let $\mathbf{g} = (g_1, \ldots, g_n)$ be a $\mathbb{F}_q$-linearly family of vectors of $\mathbb{F}_{q^m}$. The Gabidulin code $\mathcal{G}_{\mathbf{g}}(n, k, m)$ is the following code $[n, k]_{q^m}$:*

$$\left\{ P(\mathbf{g}), \deg_q P < k \right\} \text{ where } P(\mathbf{g}) \text{ denotes the evaluation of the coordinates of } \mathbf{g} \text{ by } P.$$

A generator matrix for $\mathcal{G}_{\mathbf{g}}$ is given by:

$$\mathbf{G} = \begin{pmatrix} g_1 & \cdots & g_n \\ g_1^q & \cdots & g_n^q \\ \vdots & \ddots & \vdots \\ g_1^{q^{k-1}} & \cdots & g_n^{q^{k-1}} \end{pmatrix}$$

These codes can efficiently decode up to $\lfloor \frac{n-k}{2} \rfloor$ errors [7]. They can therefore be used in combination of the McEliece cryptosystem. But the resulting scheme [8] has been attacked due to the strong algebraic structure [24].

Decoding Gabidulin codes. The algorithm employed in order to decode Gabidulin codes has been proposed in [21] and later improved in [4]. Let $\mathcal{G}_{\mathbf{g}}$ denote a Gabidulin code over $\mathbb{F}_{q^m}$ of length n and dimension k generated by the vector $\mathbf{g} \in \mathbb{F}_{q^m}^n$. The decoding problem is stated as follows.

Definition 1.1.13 (**Decoding**$(\mathbf{y}, \mathcal{G}_{\mathbf{g}}, t)$ [21])**.** *Find, if it exists,* $\mathbf{c} \in \mathcal{G}_{\mathbf{g}}$ *and* $\mathbf{e}$ *with* $\omega(\mathbf{e}) \leq t$ *such that* $\mathbf{y} = \mathbf{c} + \mathbf{e}$.

The decoding problem is solved using q-polynomial reconstruction. The reconstruction problem is defined as:

Definition 1.1.14 (**Reconstruction**$(\mathbf{y}, \mathbf{g}, k, t)$ [21])**.** *Find a tuple* (V, f) *where* V *is a non-zero q-polynomial with* $\deg_q(V) \leq t$ *and* f *is a q-polynomial with* $\deg_q(f) < k$ *such that:*

$$V(y_i) = V \circ f(g_i) \text{ with } 1 \leq i \leq n$$

When t is less than the code's decoding capacity $\lfloor (n-k)/2 \rfloor$, the solution of **Reconstruction**$(\mathbf{y}, \mathbf{g}, k, t)$ is unique. Moreover, from a solution of the q-polynomial reconstruction problem, one can get a solution to the decoding problem.

Theorem 1.2 ([21])**.** *If* (V, f) *is a solution of* **Reconstruction**$(\mathbf{y}, \mathbf{g}, k, t)$*, then* $(\mathbf{c} = f(\mathbf{g}), \mathbf{e} = \mathbf{y} - \mathbf{c})$ *is a solution of* **Decoding**$(\mathbf{y}, \mathcal{G}_{\mathbf{g}}, t)$*.*

We now consider the linearized variant of the q-polynomial reconstruction problem.

Definition 1.1.15 (**Reconstruction2**$(\mathbf{y}, \mathbf{g}, k, t)$ [21])**.** *Find a tuple* (V, N) *where* V *is a non-zero q-polynomial with* $\deg_q(V) \leq t$ *and* N *is a q-polynomial with* $\deg_q(N) \leq k + t - 1$ *such that:*

$$V(y_i) = N(g_i) \text{ with } 1 \leq i \leq n$$

When t is less than the code's decoding capacity $\lfloor (n-k)/2 \rfloor$, the two reconstruction problems are equivalent.

Theorem 1.3 ([21])**.** *If* (V, f) *is a solution of* **Reconstruction**$(\mathbf{y}, \mathbf{g}, k, t)$*, then* $(V, V \circ f)$ *is a solution of* **Reconstruction2**$(\mathbf{y}, \mathbf{g}, k, t)$*.*

If $t \leq \lfloor (n-k)/2 \rfloor$ *and if* (V, N) *is a solution of* **Reconstruction2**$(\mathbf{y}, \mathbf{g}, k, t)$*, then* (V, f) *with* f *defined as the left euclidean division of* N *by* V *in the ring of q-polynomials is a solution of* **Reconstruction**$(\mathbf{y}, \mathbf{g}, k, t)$*.*

In order to solve the **Reconstruction2**$(\mathbf{y}, \mathbf{g}, k, t)$ problem, one constructs by recurrence two pairs of q-polynomials (N_0, V_0) and (N_1, V_1) satisfying the interpolation conditions of the problem $V(y_i) = N(g_i), 1 \leq i \leq n$ at each step i and such that at least one of the pairs satisfies the final degree conditions $\deg_q(V) \leq t$ and $\deg_q(N) \leq k + t - 1$. The complete description of this algorithm can be found in [4], section 4, algorithm 5.

In a nutshell, the decoding algorithm for Gabidulin codes works as follows:

Definition 1.1.16 (**Algorithm for Decoding**$(\mathbf{y}, \mathcal{G}_{\mathbf{g}}, t)$ [21, 4])**.**

1. *Find a solution* (V, N) *of* **Reconstruction2**$(\mathbf{y}, \mathbf{g}, k, t)$
2. *Find* f *by computing the left euclidean division of* N *by* V
3. *Retrieve the codeword* $\mathbf{c}$ *by evaluating* f *in* $\mathbf{g}$

Theorem 1.4 ([4])**.** *The complexity of solving* **Decoding**$(\mathbf{y}, \mathcal{G}_{\mathbf{g}}, t)$ *by using the algorithm described in definition 1.1.16 is* $\mathcal{O}(n^2)$ *operations in* $\mathbb{F}_{q^m}$*. More precisely, the number of different operations is upper-bounded by:*

- $2n^2 - 2n + (k-1)(\frac{n-k}{2})$ *additions in* $\mathbb{F}_{q^m}$ *;*
- $2n^2 - k + (k-1)(\frac{n-k}{2})$ *multiplications in* $\mathbb{F}_{q^m}$ *;*
- $n^2 + 0.5k^2 - 2n + 1.5k^2 + (n-k)(k-1)$ *exponentiations by* q *in* $\mathbb{F}_{q^m}$ *;*
- $2n$ *divisions in* $\mathbb{F}_{q^m}$*.*

1.1.3 Difficult problems for cryptography

In this section we describe difficult problems which can be used for cryptography and discuss their complexity.

All problems are variants of the *decoding problem*, which consists of looking for the closest codeword to a given vector: when dealing with linear codes, it is readily seen that the decoding problem stays the same when one is given the *syndrome* of the received vector rather than the received vector. We therefore speak of (rank) *Syndrome Decoding* (RSD).

Definition 1.1.17 (RSD Distribution)**.** *For positive integers,* n*,* k*, and* w*, the* $\mathsf{RSD}(n, k, w)$ Distribution *chooses* $\mathbf{H} \xleftarrow{\$} \mathbb{F}_{q^m}^{(n-k)\times n}$ *and* $\mathbf{x} \xleftarrow{\$} \mathbb{F}_{q^m}^{n}$ *such that* $\omega(\mathbf{x}) = w$*, and outputs* $(\mathbf{H}, \sigma(\mathbf{x}) = \mathbf{H}\mathbf{x}^\top)$*.*

Definition 1.1.18 (Search RSD Problem)**.** *On input* $(\mathbf{H}, \mathbf{y}^\top) \in \mathbb{F}_{q^m}^{(n-k)\times n} \times \mathbb{F}_{q^m}^{(n-k)}$ *from the RSD distribution, the* Rank Syndrome Decoding Problem $\mathsf{RSD}(n, k, w)$ *asks to find* $\mathbf{x} \in \mathbb{F}_{q^m}^{n}$ *such that* $\mathbf{H}\mathbf{x}^\top = \mathbf{y}^\top$ *and* $\omega(\mathbf{x}) = w$*.*

The RSD problem has recently been proven difficult with a probabilistic reduction to the Hamming setting in [13]. For cryptography we also need a decision version of the problem, which is given in the following definition.

Definition 1.1.19 (Decision RSD Problem)**.** *On input* $(\mathbf{H}, \mathbf{y}^\top) \xleftarrow{\$} \mathbb{F}_{q^m}^{(n-k)\times n} \times \mathbb{F}_{q^m}^{(n-k)}$*, the* Decision RSD Problem $\mathsf{DRSD}(n, k, w)$ *asks to decide with non-negligible advantage whether* $(\mathbf{H}, \mathbf{y}^\top)$ *came from the RSD*(n, k, w) *distribution or the uniform distribution over* $\mathbb{F}_{q^m}^{(n-k)\times n} \times \mathbb{F}_{q^m}^{(n-k)}$*.*

Finally, as our cryptosystem will use QC-codes, we explicitly define the problem on which our cryptosystem will rely. The following definitions describe the DRSD problem in the QC configuration, and are just a combination of Def. 1.1.7 and 1.1.19. Quasi-Cyclic codes are very useful in cryptography since their compact description allows to decrease considerably the size of the keys. In particular the case $s = 2$ corresponds to double circulant codes with generator matrices of the form $(\mathbf{I}_n \mid \mathbf{A})$ for $\mathbf{A}$ a circulant matrix. Such double circulant codes have been used for almost 10 years in cryptography (cf [10]) and more recently in [22]. Quasi-cyclic codes of index 3 are also considered in [22].

Definition 1.1.20 (s-RQCSD Distribution)**.** *For positive integers n, w and s, the s-*$\mathsf{RQCSD}(n, w)$ Distribution *chooses uniformly at random a parity matrix* $\mathbf{H} \xleftarrow{\$} \mathbb{F}_{q^m}^{(sn-n)\times sn}$ *of a systematic QC code $\mathcal{C}$ of index s and rate $1/s$ (see Def. 1.1.8) together with a vector* $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_s) \xleftarrow{\$} \mathbb{F}_{q^m}^{sn}$ *such that $\omega(\mathbf{x}_i) = w$, $i = 1..s$, and outputs $(\mathbf{H}, \mathbf{H}\mathbf{x}^\top)$.*

Definition 1.1.21 ((Search) s-RQCSD Problem)**.** *For positive integers n, w, s, a random parity check matrix $\mathbf{H}$ of a systematic QC code $\mathcal{C}$ of index s and $\mathbf{y} \xleftarrow{\$} \mathbb{F}_{q^m}^{sn-n}$, the* Search s-Quasi-Cyclic RSD Problem s-$\mathsf{RQCSD}(n, w)$ *asks to find $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_s) \in \mathbb{F}_{q^m}^{sn}$ such that $\omega(\mathbf{x}_i) = w$, $i = 1..s$, and $\mathbf{y} = \mathbf{x}\mathbf{H}^\top$.*

It would be somewhat more natural to choose the parity-check matrix $\mathbf{H}$ to be made up of independent uniformly random circulant submatrices, rather than with the special form required by (7). We choose this distribution so as to make the security reduction to follow less technical. It is readily seen that, for fixed s, when choosing quasi-cyclic codes with this more general distribution, one obtains with non-negligible probability, a quasi-cyclic code that admits a parity-check matrix of the form (7). Therefore requiring quasi-cyclic codes to be systematic does not hurt the generality of the decoding problem for quasi-cyclic codes.

Assumption 1. *Although there is no general complexity result for quasi-cyclic codes, decoding these codes is considered hard by the community. There exist general attacks which uses the cyclic structure of the code [26] but these attacks have only a very limited impact on the practical complexity of the problem. The conclusion is that in practice, the best attacks are the same as those for non-circulant codes up to a small factor.*

The problem has a decisional form:

Definition 1.1.22 (Decision s-RQCSD Problem)**.** *For positive integers n, w, s, a random parity check matrix $\mathbf{H}$ of a systematic QC code $\mathcal{C}$ and $\mathbf{y} \xleftarrow{\$} \mathbb{F}_{q^m}^{sn}$, the* Decision s-Quasi-Cyclic RSD Problem s-$\mathsf{DRQCSD}(n, w)$ *asks to decide with non-negligible advantage whether $(\mathbf{H}, \mathbf{y}^\top)$ came from the s-$\mathsf{RQCSD}(n, w)$ distribution or the uniform distribution over $\mathbb{F}_{q^m}^{(sn-n)\times sn} \times \mathbb{F}_{q^m}^{(sn-n)}$.*

As for the ring-LPN problem, there is no known reduction from the search version of s-RQCSD problem to its decision version. The proof of [2] cannot be directly adapted in the quasi-cyclic case, however the best known attacks on the decision version of the problem s-RQCSD remain the direct attacks on the search version of the problem s-RQCSD.

1.1.4 Encryption and security

Encryption Scheme. An encryption scheme is a tuple of four polynomial time algorithms $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$:

- $\mathsf{Setup}(1^\lambda)$, where λ is the security parameter, generates the global parameters param of the scheme;
- $\mathsf{KeyGen}(\mathsf{param})$ outputs a pair of keys, a (public) encryption key pk and a (private) decryption key sk;
- $\mathsf{Encrypt}(\mathsf{pk}, \mathbf{m})$ outputs a ciphertext $\mathbf{c}$, on the message $\mathbf{m}$, under the encryption key pk;
- $\mathsf{Decrypt}(\mathsf{sk}, \mathbf{c})$ outputs the plaintext $\mathbf{m}$, encrypted in the ciphertext $\mathbf{c}$ or $\perp$.

Such an encryption scheme has to satisfy both *Correctness* and *Indistinguishability under Chosen Plaintext Attack* (IND-CPA) security properties.

Correctness: For every λ, every $\mathsf{param} \leftarrow \mathsf{Setup}(1^\lambda)$, every pair of keys $(\mathsf{pk}, \mathsf{sk})$ generated by KeyGen, every message $\mathbf{m}$, we should have $P[\mathsf{Decrypt}(\mathsf{sk}, \mathsf{Encrypt}(\mathsf{pk}, \mathbf{m}, \theta)) = \mathbf{m}] = 1 - \mathsf{negl}(\lambda)$ for $\mathsf{negl}(\cdot)$ a negligible function, where the probability is taken over varying randomness.

IND-CPA [15]: This notion formalized by the game depicted in Fig. 1, states that an adversary should not be able to efficiently guess which plaintext has been encrypted even if he knows it is one among two plaintexts of his choice.

In the following, we denote by $|\mathcal{A}|$ the running time of an adversary $\mathcal{A}$. The global advantage for polynomial time adversaries running in time less than t is:

$$\mathsf{Adv}_{\mathcal{E}}^{\mathsf{ind}}(\lambda, t) = \max_{|\mathcal{A}| \leq t} \mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathsf{ind}}(\lambda), \tag{12}$$

where $\mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathsf{ind}}(\lambda)$ is the advantage the adversary $\mathcal{A}$ has in winning game $\mathbf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ind}-b}(\lambda)$:

```
$\mathbf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ind}-b}(\lambda)$
1. $\mathsf{param} \leftarrow \mathsf{Setup}(1^\lambda)$
2. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{param})$
3. $(\mathbf{m}_0, \mathbf{m}_1) \leftarrow \mathcal{A}(\texttt{FIND} : \mathsf{pk})$
4. $\mathbf{c}^* \leftarrow \mathsf{Encrypt}(\mathsf{pk}, \mathbf{m_b}, \theta)$
5. $b' \leftarrow \mathcal{A}(\texttt{GUESS} : \mathbf{c}^*)$
6. RETURN $b'$
```

Figure 1: Game for the IND-CPA security of an asymmetric encryption scheme.

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{E},\mathcal{A}}(\lambda) = \left|\Pr[\mathbf{Exp}^{\mathsf{ind}-1}_{\mathcal{E},\mathcal{A}}(\lambda) = 1] - \Pr[\mathbf{Exp}^{\mathsf{ind}-0}_{\mathcal{E},\mathcal{A}}(\lambda) = 1]\right| . \tag{13}$$

IND-CPA and IND-CCA2: Note that the standard security requirement for a public key cryptosystem is IND-CCA2, *indistinguishability against adaptive chosen-ciphertext attacks*, and not just IND-CPA. The main difference is that for IND-CCA2 indistinguishability must hold even if the attacker is given a *decryption oracle* first when running the `FIND` algorithm and also when running the `GUESS` algorithm (but cannot query the oracle on the challenge ciphertext $\boldsymbol{c}^*$). We do not present the associated formal game and definition as an existing (and inexpensive) transformation can be used [17] for our scheme to pass from IND-CPA to IND-CCA2.

In [17] Hofheinz et al. present a generic transformation that takes into account decryption errors and can be applied directly to our scheme. Roughly, their construction provides a way to convert a guarantee against passive adversaries into indistinguishability against active ones by turning a public key cryptosystem into a KEM-DEM. The tightness (the quality factor) of the reduction depends on the ciphertext distribution. Regarding our scheme, random words only have a negligible (in the security parameter) probability of being valid ciphertexts. In other words, the γ-spreadness factor of [17] is small enough so that there is no loss between the IND-CPA security of our public key cryptosystem and the IND-CCA2 security of the KEM-DEM version.

The security reduction is tight in the random oracle model and does not require any supplemental property from our scheme as we have the IND-CPA property (instead of just a weaker property called *One-Wayness*). Let us denote by $\mathsf{Encrypt}(\mathsf{pk}, \mathbf{m}, \theta)$ the encryption function defined in Fig. 3 that uses randomness θ to generate uniformly random values $\mathbf{r}_1$, $\mathbf{r}_2$, and $\mathbf{e}$. The idea of [17] transformation is to de-randomize the encryption function $\mathsf{Encrypt}(\mathsf{pk}, \mathbf{m}, \theta)$ by using a hash function $\mathcal{G}$ and do a deterministic encryption of $\mathbf{m}$ by calling $c = \mathsf{Encrypt}(\mathsf{pk}, \mathbf{m}, \mathcal{G}(\mathbf{m}))$. The ciphertext is sent together with a hash $K = \mathcal{H}(\mathbf{c}, \mathbf{m})$ that ties the ciphertext to the plaintext. The receiver then decrypts $\mathbf{c}$ into $\mathbf{m}$, checks the hash value, and uses again the deterministic encryption to check that $\mathbf{c}$ is indeed *the* ciphertext associated to $\mathbf{m}$.

As the reduction is tight we do not need to change our parameters when we pass from IND-CPA to IND-CCA2. From a computational point of view, the overhead for the sender is two hash calls and for the receiver it is two hash calls and an encrypt call. From a communication point of view the overhead is the bitsize of a hash (or two if the reduction must hold in the Quantum Random Oracle Model, see [17] for more details).

1.2 Presentation of the scheme

In this section, we describe our proposal: RQC. We begin with the PKE version (RQC.PKE), then describe the transformation of [17] to obtain a KEM-DEM that achieves IND-CCA2 (RQC.KEM). Finally, we discuss an hybrid encryption scheme using NIST standard conversion techniques (RQC.HE). Parameter sets can be found in Sec. 1.3.

1.2.1 Public key encryption version (RQC.PKE)

Presentation of the scheme. RQC uses two types of codes: a decodable $[n, k]$ code $\mathcal{C}$, generated by $\mathbf{G} \in \mathbb{F}^{k \times n}$ and which can correct at least δ errors via an efficient algorithm $\mathcal{C}.\mathsf{Decode}(\cdot)$ (*e.g.* a Gabidulin code); and a random double-circulant $[2n, n]$ code, of parity-check matrix $(\mathbf{1}, \mathbf{h})$. The four polynomial-time algorithms constituting our scheme are depicted in Fig. 3.

- $\mathsf{Setup}(1^\lambda)$: generates and outputs the global parameters $\mathsf{param} = (n, k, \delta, w, w_{\mathbf{r}}, w_{\mathbf{e}})$.
- $\mathsf{KeyGen}(\mathsf{param})$: samples $\mathbf{h} \overset{\$}{\leftarrow} \mathcal{R}$, the generator matrix $\mathbf{G} \in \mathbb{F}_{q^m}^{k \times n}$ of $\mathcal{C}$, $\mathsf{sk} = (\mathbf{x}, \mathbf{y}) \overset{\$}{\leftarrow} \mathcal{R}^2$ such that $\omega(\mathbf{x}) = \omega(\mathbf{y}) = w$, sets $\mathsf{pk} = (\mathbf{h}, \mathbf{s} = \mathbf{x} + \mathbf{h} \cdot \mathbf{y})$, and returns $(\mathsf{pk}, \mathsf{sk})$.
- $\mathsf{Encrypt}(\mathsf{pk}, \mathbf{m})$: generates $\mathbf{e} \overset{\$}{\leftarrow} \mathcal{R}$, $\mathbf{r} = (\mathbf{r}_1, \mathbf{r}_2) \overset{\$}{\leftarrow} \mathcal{R}^2$ such that $\omega(\mathbf{e}) = w_{\mathbf{e}}$ and $\omega(\mathbf{r}_1) = \omega(\mathbf{r}_2) = w_{\mathbf{r}}$, sets $\mathbf{u} = \mathbf{r}_1 + \mathbf{h} \cdot \mathbf{r}_2$ and $\mathbf{v} = \mathbf{m}\mathbf{G} + \mathbf{s} \cdot \mathbf{r}_2 + \mathbf{e}$, returns $\mathbf{c} = (\mathbf{u}, \mathbf{v})$.
- $\mathsf{Decrypt}(\mathsf{sk}, \mathbf{c})$: returns $\mathcal{C}.\mathsf{Decode}(\mathbf{v} - \mathbf{u} \cdot \mathbf{y})$.

Figure 2: Description of our proposal RQC.PKE.

Notice that the generator matrix $\mathbf{G}$ of the code $\mathcal{C}$ is publicly known, so the security of the scheme and the ability to decrypt do not rely on the knowledge of the error correcting code $\mathcal{C}$ being used.

Correctness. The correctness of our new encryption scheme clearly relies on the decoding capability of the code $\mathcal{C}$. Specifically, assuming $\mathcal{C}.\mathsf{Decode}$ correctly decodes $\mathbf{v} - \mathbf{u} \cdot \mathbf{y}$, we have:

$$\mathsf{Decrypt}\,(\mathrm{sk}, \mathsf{Encrypt}\,(\mathrm{pk}, \mathbf{m})) = \mathbf{m}. \tag{14}$$

And $\mathcal{C}.\mathsf{Decode}$ correctly decodes $\mathbf{v} - \mathbf{u} \cdot \mathbf{y}$ whenever

$$\omega\left(\mathbf{s} \cdot \mathbf{r}_2 - \mathbf{u} \cdot \mathbf{y} + \mathbf{e}\right) \leq \delta \tag{15}$$

$$\omega\left((\mathbf{x} + \mathbf{h} \cdot \mathbf{y}) \cdot \mathbf{r}_2 - (\mathbf{r}_1 + \mathbf{h} \cdot \mathbf{r}_2) \cdot \mathbf{y} + \mathbf{e}\right) \leq \delta \tag{16}$$

$$\omega\left(\mathbf{x} \cdot \mathbf{r}_2 - \mathbf{r}_1 \cdot \mathbf{y} + \mathbf{e}\right) \leq \delta \tag{17}$$

In contrast to HQC, there is no decryption failure, or to be more accurate, the probability that a decryption failure occurs is null. More details are provided at the beginning of Sec. 1.3.

1.2.2 KEM/DEM version (RQC.KEM)

Let $\mathcal{E}$ be an instance of the RQC cryptosystem as described above. Let $\mathcal{G}$, $\mathcal{H}$, and $\mathcal{K}$ be hash functions, typically SHA512 as advised by NIST[1]. The KEM-DEM version of the RQC

---

[1] See Dustin Moody's mail entitled "new FAQ question" on PQC-forum (20/07/2017 – 12:58)

cryptosystem is defined as follows:

- $\mathsf{Setup}(1^\lambda)$: as before, except that the plaintext space has size $k \times m \geq 256$ as required by NIST.
- $\mathsf{KeyGen}(\mathsf{param})$: exactly as before.
- $\mathsf{Encapsulate}(\mathsf{pk})$: generate $\mathbf{m} \xleftarrow{\$} \mathbb{F}_{q^m}^k$ (this will serve as a seed to derive the shared key). Derive the randomness $\theta \leftarrow \mathcal{G}(\mathbf{m})$. Generate the ciphertext $c \leftarrow (\mathbf{u}, \mathbf{v}) = \mathcal{E}.\mathsf{Encrypt}(\mathsf{pk}, \mathbf{m}, \theta)$, and derive the symmetric key $K \leftarrow \mathcal{K}(\mathbf{m}, \mathbf{c})$. Let $\mathbf{d} \leftarrow \mathcal{H}(\mathbf{m})$, and send $(\mathbf{c}, \mathbf{d})$.
- $\mathsf{Decapsulate}(\mathsf{sk}, \mathbf{c}, \mathbf{d})$: Decrypt $\mathbf{m}' \leftarrow \mathcal{E}.\mathsf{Decrypt}(\mathsf{sk}, \mathbf{c})$, compute $\theta' \leftarrow \mathcal{G}(\mathbf{m}')$, and (re-)encrypt $\mathbf{m}'$ to get $\mathbf{c}' \leftarrow \mathcal{E}.\mathsf{Encrypt}(\mathsf{pk}, \mathbf{m}', \theta')$. If $\mathbf{c} \neq \mathbf{c}'$ or $\mathbf{d} \neq \mathcal{H}(\mathbf{m}')$ then abort. Otherwise, derive the shared key $K \leftarrow \mathcal{K}(\mathbf{m}, \mathbf{c})$.

Figure 3: Description of our proposal RQC.KEM.

According to [17], the KEM-DEM version of RQC is IND-CCA2. More details regarding the tightness of the reduction are provided at the end of Sec. 1.3.

Security concerns and implementation details. Notice that while NIST only recommends SHA512 as a hash function (or TupleHash256 for hardware efficiency purposes), the transformation of [17] would be dangerous – at least in our setting – if one sets $\mathcal{G} = \mathcal{H}$. Indeed, publishing the randomness $\theta = \mathcal{G}(\mathbf{m}) = \mathcal{H}(\mathbf{m}) = \mathbf{d}$ used to generate $\mathbf{r}_1$, $\mathbf{r}_2$, and $\mathbf{e}$, would allow one to retrieve $\mathbf{s}$, the secret key of $\mathcal{E}$.

We therefore suggest to use a pseudo-random function for $\mathcal{G}$, such as an AES-based seed expander, and SHA512 for $\mathcal{H}$.

1.2.3 A hybrid encryption scheme (RQC.HE)

While NIST claimed that they will be using generic transformations to convert any IND-CCA2 KEM into an IND-CCA2 PKE, no detail on these conversions have been provided. We therefore refer to RQC.HE to designate the PKE scheme resulting from applying a generic conversion to RQC.KEM.

1.3 Parameters

Error distribution and decoding algorithm: no decryption failure. The case of the rank metric is much simpler than for the Hamming metric. Indeed in that case the decryption algorithm of our cryptosystem asks to decode an error $\mathbf{e}' = \mathbf{x} \cdot \mathbf{r}_2 - \mathbf{r}_1 \cdot \mathbf{y} + \mathbf{e}$ where the words $\mathbf{x}$ and $\mathbf{y}$ (resp. $\mathbf{r}_1$ and $\mathbf{r}_2$)) have rank weight w (resp. $w_\mathbf{r}$). Unlike the Hamming metric weight, the rank weight of the vector $\mathbf{x} \cdot \mathbf{r}_2 - \mathbf{r}_1 \cdot \mathbf{y}$ is almost always $ww_\mathbf{r}$ and is in any case bounded from above by $ww_\mathbf{r}$. In particular, with a strong probability, the rank weight of $\mathbf{x}\cdot\mathbf{r}_2-\mathbf{r}_1\cdot\mathbf{y}$ is the same as the rank weight of $\mathbf{x}\cdot\mathbf{r}_2$ since $\mathbf{x}$ and $\mathbf{y}$ share the same rank support, as do $\mathbf{r}_1$ and $\mathbf{r}_2$. We consider the additional error $\mathbf{e}$ of rank $w_\mathbf{e}=w_\mathbf{r}$ with same error support as $\mathbf{r}_1$ and $\mathbf{r}_2$. So that overall the error $\mathbf{e}'$ to decode for decryption has a rank weight upper bounded by $(w+1)w_\mathbf{r}$.

Now it is possible to optimize a little bit the weight of $\mathbf{e}'$ by considering that the support of the secret vector $(\mathbf{x},\mathbf{y})$ is a random subspace of $\mathbb{F}_{q^m}$ of dimension w containing 1, indeed in that case the weight of $\mathbf{e}'$ is upper bounded by $ww_\mathbf{r}$ since the support of $\mathbf{e}$ is included in the product of the supports of $(\mathbf{x},\mathbf{y})$ and $(\mathbf{r}_1,\mathbf{r}_2)$. This does not modify the security proof, and impacts only the value of w in the choice of parameters.

For decoding, we consider Gabidulin $[n,k]$ codes over $\mathbb{F}_{q^n}$, which can decode $\frac{n-k}{2}$ rank errors and choose our parameters such that $ww_\mathbf{r}\leq\frac{n-k}{2}$, so that, unlike the Hamming metric case, *there is no decryption failure.*

Parameters and tightness of the reduction. The practical security of the scheme relies on the 2-DRQCSD problem for the public key, for a small weight vector of weight $w=\omega(\mathbf{x})=\omega(\mathbf{y})$ with $w=\mathcal{O}(\sqrt{n})$. The IND-CPA security of the scheme could be reduced to the 3-DRQCSD problem, decoding a random quasi-cyclic $[3n,n]$ code for a small weight vector $(\mathbf{r}_1,\mathbf{e},\mathbf{r}_2)$. In the proof, the error vectors $\mathbf{r}_1$ and $\mathbf{r}_2$ share the same error support E of dimension $w_\mathbf{r}$, for the encryption part the error support of $\mathbf{e}$ can also be taken as E, so that the problem is tightly reduced to the 3-DRQCSD problem for rank metric with weight $w_\mathbf{r}$, since all three vectors $\mathbf{r}_1,\mathbf{r}_2$ and $\mathbf{e}$ have the same error support E of dimension $w_\mathbf{r}$. In that case the attacker wants to decode a $[3n,n]$ rank metric code, the best known attack is described in [12, 3]. Since on one hand the attacker wants to attack a length $2n$ code and on the other hand to attack a length $3n$ code, which is easier, we consider different weights for the secret key $\mathbf{x},\mathbf{y}$ of weight w and for the random chosen values for the encryption $\mathbf{r}_1,\mathbf{e},\mathbf{r}_2$ of weight $w_\mathbf{r}=w_\mathbf{e}$, typically we chose $w\approx\frac{2}{3}w_\mathbf{r}$. For the secret key, we consider $1\in \mathit{Support}(\mathbf{x},\mathbf{y})$, now since finding a small weight codeword of weight w with support containing 1 is harder than finding a small weight vector of weight $w-1$, we consider $w-1$ for the security reduction to the 2-DRQCSD problem, and the weight $w_\mathbf{r}=w_\mathbf{e}$ is chosen according to the 3-DRQCSD problem and the best known attacks of [12, 3], whose complexity is given in section II-D. The best quantum attacks on the rank metric problems follow [11], in that case there is square root gain on the probabilistic part of the attack (details are given in [11]).

Remark 1.5. *The system is based on cyclic codes, which means considering polynomials modulo x^n-1, interestingly enough, and only in the case of the rank metric, the construction remains valid when considering not only polynomials modulo x^n-1 but also modulo a polynomial with coefficient in the base field $\mathbb{F}_q$. Indeed in that case the modulo does not change the rank weight of a codeword. Such a variation on the scheme may be interesting to avoid potential structural attacks which may use the factorization of the quotient polynomial for the considered polynomial ring.*

Choice of parameters: overall the parameters proposed in Tab. 1 correspond to tight reduction for generic instances of the 2-DRQCSD and 3-DRQCSD problems in the rank metric. Parameters are chosen such that $1 \in \text{Supp}(\mathbf{x}, \mathbf{y})$, the vectors $\mathbf{r}_1, \mathbf{r}_2$ and $\mathbf{e}$ have the same random support of dimension $w_{\mathbf{r}} = w_{\mathbf{e}}$. The value of n is chosen so that $X^n - 1$ has up to 3 factors of high degree (except $X - 1$) in $\mathbb{F}_q[X]$ (typically n is chosen primitive modulo q). The decoding Gabidulin code has length n, dimension k over $\mathbb{F}_{q^m}$ and corrects errors of weight up to $(n-k)/2 = ww_{\mathbf{r}}$. The resulting public key, secret key, ciphertext and shared secret sizes are given in Tab. 2. One may use seeds to shorten keys thus obtaining sizes presented in Tab. 3. The aforementioned sizes are the ones used in our reference implementation except that we also concatenate the public key within the secret key in order to respect the NIST API.

	RQC Cryptosystem Parameters						
Instance	q	m	n	k	w	$w_{\mathbf{r}} = w_{\mathbf{e}}$	Security
RQC-I	2	89	67	7	5	6	128
RQC-II	2	113	97	13	6	7	192
RQC-III	2	139	101	5	6	8	256

Table 1: Parameter sets for RQC. The security is expressed in bits.

Instance	pk size	sk size	ct size	ss size	Security
RQC-I	1491	1491	1555	64	128
RQC-II	2741	2741	2805	64	192
RQC-III	3510	3510	3574	64	256

Table 2: Resulting theoretical sizes in bytes for RQC. The public key pk is composed of ($\mathbf{h}$, $\mathbf{s}$) and has size $2nm$. The secret key sk is composed of ($\mathbf{x}$, $\mathbf{y}$) and has size $2nm$. The ciphertext ct is composed of ($\mathbf{u}$, $\mathbf{v}$, $\mathbf{d}$) and has size $2nm + 64$. The shared secret ss is composed of K and has size 64 (SHA512 output size). The security is expressed in bits.

Computational Cost. The encryption cost corresponds to a matrix-vector product over $\mathbb{F}_{q^m}$, for a multiplication cost of elements of $\mathbb{F}_{q^m}$ in $m \log(m) \log(\log(m))$, we obtain an encryption complexity in $\mathcal{O}\left(n^2 m \log(m) \log(\log(m))\right)$. The decryption cost is also a matrix-vector multiplication plus the decoding cost of the Gabidulin codes, both have the complexities in $\mathcal{O}\left(n^2 m \log(m) \log(\log(m))\right)$.

2 Performance Analysis

In this section, we provide concrete performance measures of our implementation. For each parameter set, results have been obtained by running 100,000 random instances and com-

Instance	`pk` size	`sk` size	`ct` size	`ss` size	Security
RQC-I	786	40	1556	64	128
RQC-II	1411	40	2806	64	192
RQC-III	1795	40	3574	64	256

Table 3: Resulting sizes in bytes for RQC using NIST seed expander initialized with 40 bytes long seeds. The public key `pk` is composed of ($\mathbf{seed1}$, $\mathbf{s}$) and has size $40 + nm$. The secret key `sk` is composed of ($\mathbf{seed2}$) and has size 40. The ciphertext `ct` is composed of ($\mathbf{u}$, $\mathbf{v}$, $\mathbf{d}$) and has size $2nm + 64$. The shared secret `ss` is composed of K and has size 64 (SHA512 output size). The security is expressed in bits.

puting their average execution time. The benchmarks have been performed on a machine running Ubuntu 16.04 LTS. The latter has 32GB of memory and an Intel® Core™ i7-4770 CPU @ 3.4GHz for which the Hyper-Threading, Turbo Boost and SpeedStep features were disabled. The scheme have been compiled with `gcc` (version 7.2.0) using the compilation flags `-O3 -std=c99 -pedantic`. The following third party libraries have been used: `openssl` (version 1.1.0f), `gmp` (version 6.1.2) and `mpfq` (version 1.1) [14].

2.1 Reference Implementation

The performances of our reference implementation on the aforementioned benchmark platform are described in Tab. 4 (timings in ms) and Tab. 5 (millions of CPU cycles required).

Instance	KeyGen	Encrypt	Decrypt
RQC-I	0.23	0.58	1.56
RQC-II	0.52	1.65	4.25
RQC-III	0.83	1.90	5.29

Table 4: Timings (in ms) of the reference implementation for different instances of RQC.

Instance	KeyGen	Encrypt	Decrypt
RQC-I	0.79	1.97	5.30
RQC-II	1.76	5.60	14.46
RQC-III	2.82	6.46	18.00

Table 5: Millions of cycles of the reference implementation for different instances of RQC.

2.2 Optimized Implementation

No optimized implementation has been provided. As a consequence, the folders `Optimized_Implementation/` and `Reference_Implementation/` are identical. Additional implementation (optimized variant using vectorization, constant-time implementation...) might be provided later.

3 Known Answer Test Values

Known Answer Test (KAT) values have been generated using the script provided by the NIST. They are available in the folder `KAT/Reference_Implementation/`. As mentioned in Sec. 2.2, since the reference and optimized implementations are identical, `KAT/Optimized_Implementation/` is just a copy of `KAT/Reference_Implementation/`.

In addition, we provide, for each parameter set, an example with *intermediate values* in the folder `KAT/Reference_Implementation/`.

Notice that one can generate the aforementioned test files using respectively the `kat` and `verbose` modes of our implementation. The procedure to follow in order to do so is detailed in the technical documentation.

4 Security

In this section we prove the security of our encryption scheme viewed as a PKE scheme (IND-CPA). The security of the KEM/DEM version is provided by the transformation described in [17], and the tightness of the reduction provided by this transformation has been discussed at the end of Sec. 1.1.4.

Theorem 4.1. *The scheme presented above is* IND-CPA *under the* 2-*DRQCSD and* 3-*DRQCSD assumptions.*

Proof. To prove the security of the scheme, we are going to build a sequence of games transitioning from an adversary receiving an encryption of message $\mathbf{m}_0$ to an adversary receiving an encryption of a message $\mathbf{m}_1$ and show that if the adversary manages to distinguish one from the other, then we can build a simulator breaking the DRQCSD assumption, for QC codes of index 2 or 3 (codes with parameters $[2n, n]$ or $[3n, n]$), and running in approximately the same time.

Game $\mathbf{G}_1$: This is the real game, which we can state algorithmically as follows:

```
$\mathbf{Game}^1_{\mathcal{E},\mathcal{A}}(\lambda)$
1. $\mathsf{param} \leftarrow \mathsf{Setup}(1^\lambda)$
2. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{param})$ with $\mathsf{pk} = \big(\mathbf{h}, \mathbf{s} = \mathsf{sk} \cdot \mathbf{h}^\top\big)$
3. $(\mathbf{m}_0, \mathbf{m}_1) \leftarrow \mathcal{A}(\mathtt{FIND} : \mathsf{pk})$
```

```
4. $\mathbf{c}^* \leftarrow \mathsf{Encrypt}(\mathsf{pk}, \mathbf{m_0}, \theta)$
5. $\mathbf{b}' \leftarrow \mathcal{A}(\texttt{GUESS} : \mathbf{c}^*)$
6. RETURN $\mathbf{b}'$
```

Game $\mathbf{G}_2$: In this game we start by forgetting the decryption key sk, and taking $\mathbf{s}$ at random, and then proceed honestly:

```
$\mathbf{Game}^2_{\mathcal{E},\mathcal{A}}(\lambda)$
1. $\mathsf{param} \leftarrow \mathsf{Setup}(1^\lambda)$
2a. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{param})$ with $\mathsf{pk} = \left(\mathbf{h}, \mathbf{s} = \mathsf{sk} \cdot \mathbf{h}^\top\right)$
2b. $\mathbf{s} \overset{\$}{\leftarrow} \mathcal{R}$
2c. $(\mathsf{pk}, \mathsf{sk}) \leftarrow ((\mathbf{h}, \mathbf{s}), \mathbf{0})$
3. $(\mathbf{m}_0, \mathbf{m}_1) \leftarrow \mathcal{A}(\texttt{FIND} : \mathsf{pk})$
4. $\mathbf{c}^* \leftarrow \mathsf{Encrypt}(\mathsf{pk}, \mathbf{m_0}, \theta)$
5. $\mathbf{b}' \leftarrow \mathcal{A}(\texttt{GUESS} : \mathbf{c}^*)$
6. RETURN $\mathbf{b}'$
```

The adversary has access to pk and $\boldsymbol{c}^*$. As he has access to pk and the $\mathsf{Encrypt}$ function, anything that is computed from pk and $\boldsymbol{c}^*$ can also be computed from just pk. Moreover, the distribution of $\boldsymbol{c}^*$ is independent of the game we are in, and therefore we can suppose the only input of the adversary is pk. Suppose he has an algorithm $\mathcal{D}_\lambda$, taking pk as input, that distinguishes with advantage ϵ Game $\mathbf{G}_1$ and Game $\mathbf{G}_2$, for some security parameter λ. Then he can also build an algorithm $\mathcal{D}'_{\mathcal{E},\mathcal{D}_\lambda}$ which solves the 2-$\mathsf{DRQCSD}(n, \omega)$ problem for parameters (n, ω) resulting from $\mathsf{Setup}(\lambda)$, with the same advantage ϵ, when given as input a challenge $(\mathbf{H}, \mathbf{y}^\top) \in \mathbb{F}_{q^m}^{n \times 2n} \times \mathbb{F}_{q^m}^n$.

```
$\mathbf{D}'_{\mathcal{E},\mathcal{D}_\lambda}((\mathbf{H}, \mathbf{y}^\top))$
1. Set $\mathsf{param} \leftarrow \mathsf{Setup}(\lambda)$ and get $\mathbf{G}$ from $\mathsf{KeyGen}(\mathsf{param})$
2. $\mathsf{pk} \leftarrow (\mathbf{h}, \mathbf{y})$
2. $b' \leftarrow \mathcal{D}_\lambda(\mathsf{pk})$
4. If $b' == 0$ output RQCSD
5. If $b' == 1$ output UNIFORM
```

Note that if we define pk as $(\mathbf{h}, \mathbf{y})$ with $\mathbf{G}$ generated by $\mathsf{KeyGen}(n, k, \delta, \omega)$ and $(\mathbf{H}, \mathbf{y}^\top)$ from a 2-$\mathsf{RQCSD}(n, \omega)$ distribution pk follows exactly the same distribution as in Game $\mathbf{G}_1$. On the other hand if $(\mathbf{H}, \mathbf{y}^\top)$ comes from a uniform distribution, pk follows exactly the same distribution as in Game $\mathbf{G}_2$. Thus we have

$$Pr\left[\mathbf{D}'_{\mathcal{E},\mathcal{D}_\lambda}((\mathbf{h}, \mathbf{y}^\top)) = \texttt{RQCSD} | (\mathbf{h}, \mathbf{y}^\top) \leftarrow 2\text{-}\mathsf{RQCSD}(n, \omega)\right] = Pr\left[\mathbf{D}_\lambda(\mathsf{pk}) = 0 | \mathsf{pk} \text{ from } \mathbf{Game}^0_{\mathcal{E},\mathcal{A}}(\lambda)\right]$$

and

$$Pr\left[\mathbf{D}'_{\mathcal{E},\mathcal{D}_\lambda}((\mathbf{h}, \mathbf{y}^\top)) = \texttt{UNIFORM} | (\mathbf{h}, \mathbf{y}^\top) \leftarrow 2\text{-}\mathsf{RQCSD}(n, \omega)\right] = Pr\left[\mathbf{D}_\lambda(\mathsf{pk}) = 1 | \mathsf{pk} \text{ from } \mathbf{Game}^0_{\mathcal{E},\mathcal{A}}(\lambda)\right]$$

And similarly when $(\mathbf{h}, \mathbf{y}^\top)$ is uniform the probabilities of $\mathcal{D}'_{\mathcal{E},\mathcal{D}_\lambda}$ outputs match those of $\mathcal{D}_\lambda$ when pk is from $\mathbf{Game}^1_{\mathcal{E},\mathcal{A}}(\lambda)$. The advantage of $\mathcal{D}'_{\mathcal{E},\mathcal{D}_\lambda}$ is therefore equal to the advantage of $\mathcal{D}_\lambda$.

Game $\mathbf{G}_3$: Now that we no longer know the decryption key, we can start generating random ciphertexts. So instead of picking correctly weighted $\mathbf{r}_1, \mathbf{r}_2, \mathbf{e}$, the simulator now picks random vectors in the full space.

```
$\mathbf{Game}^3_{\mathcal{E},\mathcal{A}}(\lambda)$
1. $\mathsf{param} \leftarrow \mathsf{Setup}(1^\lambda)$
2a. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{param})$ with $\mathsf{pk} = \left(\mathbf{h}, \mathbf{s} = \mathsf{sk} \cdot \mathbf{h}^\top\right)$
2b. $\mathbf{s} \overset{\$}{\leftarrow} \mathcal{R}$
2c. $(\mathsf{pk}, \mathsf{sk}) \leftarrow ((\mathbf{h}, \mathbf{s}), \mathbf{0})$
3. $(\mathbf{m}_0, \mathbf{m}_1) \leftarrow \mathcal{A}(\mathtt{FIND} : \mathsf{pk})$
4a. Use randomness $\theta$ to generate $\mathbf{e} \overset{\$}{\leftarrow} \mathcal{R}$, $\mathbf{r} = (\mathbf{r}_1, \mathbf{r}_2) \overset{\$}{\leftarrow} \mathcal{R}^2$ uniformly at random
4b. $\mathbf{u}^\top \leftarrow \mathbf{H}\mathbf{r}^\top$ and $\mathbf{v} \leftarrow \mathbf{m}_0\mathbf{G} + \mathbf{s} \cdot \mathbf{r}_2 + \mathbf{e}$
4c. $\mathbf{c}^* \leftarrow (\mathbf{u}, \mathbf{v})$
5. $\mathbf{b}' \leftarrow \mathcal{A}(\mathtt{GUESS} : \mathbf{c}^*)$
6. RETURN $\mathbf{b}'$
```

As we have

$$(\mathbf{u}, \mathbf{v} - \mathbf{m}_0\mathbf{G})^\top = \begin{pmatrix} \mathbf{I}_n & \mathbf{0} & \mathbf{rot}(\mathbf{h}) \\ \mathbf{0} & \mathbf{I}_n & \mathbf{rot}(\mathbf{s}) \end{pmatrix} \cdot (\mathbf{r}_1, \mathbf{e}, \mathbf{r}_2)^\top,$$

the difference between Game $\mathbf{G}_2$ and Game $\mathbf{G}_3$ is that in the former

$$\left(\begin{pmatrix} \mathbf{I}_n & \mathbf{0} & \mathbf{rot}(\mathbf{h}) \\ \mathbf{0} & \mathbf{I}_n & \mathbf{rot}(\mathbf{s}) \end{pmatrix}, (\mathbf{u}, \mathbf{v} - \mathbf{m}_0\mathbf{G})^\top \right)$$

follows the 3-RQCSD distribution (for a $2n \times 3n$ QC matrix of index 3), and in the latter it follows a uniform distribution (as $\boldsymbol{r}_1$ and $\mathbf{e}$ are uniformly distributed and independently chosen One-Time Pads).

Note that an adversary is not able to obtain $\boldsymbol{c}^*$ from pk any more, as depending on which game we are $\boldsymbol{c}^*$ is generated differently. The input of a game distinguisher will therefore be $(\mathsf{pk}, \boldsymbol{c}^*)$. As it must interact with the challenger as usually we suppose it has two access modes `FIND` and `GUESS` to process first pk and later $\boldsymbol{c}^*$.

Suppose the adversary is able to distinguish Game $\mathbf{G}_2$ and Game $\mathbf{G}_3$, with a distinguisher $\mathcal{D}_\lambda$, which takes as input $(\mathsf{pk}, \boldsymbol{c}^*)$ and outputs a guess $b' \in \{1, 2\}$ of the game we are in.

Again, we can build a distinguisher $\mathcal{D}'_{\mathcal{E},\mathcal{D}_\lambda}$ that will break the 3-$\mathsf{DRQCSD}(n, \omega)$ assumption for parameters (n, ω) from $\mathsf{Setup}(1^\lambda)$ with the same advantage as the game distinguisher, when given an input $(H, y^\top) \in \mathbb{F}_{q^m}^{2n\times 3n} \times \mathbb{F}_{q^m}^{2n}$. In the 3-$\mathsf{DRQCSD}(n,\omega)$ problem, matrix H is assumed to be of the form

$$\begin{pmatrix} \mathbf{I}_n & \mathbf{0} & \mathbf{rot}(\mathbf{a}) \\ \mathbf{0} & \mathbf{I}_n & \mathbf{rot}(\mathbf{b}) \end{pmatrix}.$$

In order to use explicitly $\mathbf{a}$ and $\mathbf{b}$ we note the matrix $\mathbf{H}_{\mathbf{a},\mathbf{b}}$ instead of just $\mathbf{H}$. We will also note $\mathbf{y} = (\mathbf{y_1}, \mathbf{y_2})$.

```
$\mathbf{D}'_{\mathcal{E},\mathcal{D}_\lambda}((\mathbf{H}_{\mathbf{a},\mathbf{b}}, (\mathbf{y_1}, \mathbf{y_2})^\top))$
1. $\mathsf{param} \leftarrow \mathsf{Setup}(1^\lambda)$
2a. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{param})$ with $\mathsf{pk} = \big(\mathbf{G}, \mathbf{Q}, \mathbf{s} = \mathsf{sk}\cdot \mathbf{h}^\top\big)$
2b. $(\mathsf{pk}, \mathsf{sk}) \leftarrow ((\mathbf{G}, (\mathbf{I_n}\ \mathbf{rot}(\mathbf{a})), \mathbf{b}), \mathbf{0})$
3. $(\mathbf{m}_0, \mathbf{m}_1) \leftarrow \mathcal{D}_\lambda(\mathtt{FIND} : \mathsf{pk})$
4. $\mathbf{u} \leftarrow \mathbf{y_1}$, $\mathbf{v} \leftarrow \mathbf{m}_0\mathbf{G} + \mathbf{y_2}$ and $\boldsymbol{c}^* \leftarrow (\mathbf{u}, \mathbf{v})$
5. $b' \leftarrow \mathbf{D}_\lambda(\mathtt{GUESS} : \mathbf{c}^*)$
4. If $\mathbf{b}' == \mathbf{1}$ output RQCSD
5. If $\mathbf{b}' == \mathbf{2}$ output UNIFORM
```

The distribution of pk is unchanged with respect to the games as the first matrix is from KeyGen, the second matrix follows the same distribution as in KeyGen, and the vectors $\mathbf{b}$ and $\mathbf{s}$ are both uniformly chosen. If $(\mathbf{H}_{\mathbf{a},\mathbf{b}}, (\mathbf{y_1}, \mathbf{y_2})^\top)$ follows the 3-$\mathsf{DRQCSD}(n,\omega)$ distribution, then

$$(\mathbf{y_1}, \mathbf{y_2})^\top = \begin{pmatrix} \mathbf{I}_n & \mathbf{0} & \mathbf{rot}(\mathbf{a}) \\ \mathbf{0} & \mathbf{I}_n & \mathbf{rot}(\mathbf{b}) \end{pmatrix} \cdot (\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3})^\top$$

with $\omega(\mathbf{x}_1) = \omega(\mathbf{x}_2) = \omega(\mathbf{x}_3) = \omega$. Thus, $\boldsymbol{c}^*$ follows the same distribution as in Game $\mathbf{G}_2$. If $(\mathbf{H}_{\mathbf{a},\mathbf{b}}, (\mathbf{y_1}, \mathbf{y_2})^\top)$ follows an uniform distribution, then $\boldsymbol{c}^*$ follows the same distribution as in Game $\mathbf{G}_3$. We obtain therefore the same equalities for the output probabilities of $\mathcal{D}'_{\mathcal{E},\mathcal{D}_\lambda}$ and $\mathcal{D}_\lambda$ as with the previous games and therefore the advantages of both distinguishers are equal.

Game $\mathbf{G}_4$: We now encrypt the other plaintext. We chose $\mathbf{r}'_1, \mathbf{r}'_2, \mathbf{e}'$ uniformly and set $\mathbf{u}^\top = \mathbf{h}\boldsymbol{r}'^\top$ and $\mathbf{v} = \mathbf{m}_1\mathbf{G} + \mathbf{s}\cdot\mathbf{r}'_2 + \mathbf{e}'$. This is the last game we describe explicitly, since, even if it is a mirror of Game $\mathbf{G}_3$, it involves a new proof.

```
$\mathbf{Game}^4_{\mathcal{E},\mathcal{A}}(\lambda)$
1. $\mathsf{param} \leftarrow \mathsf{Setup}(1^\lambda)$
2a. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{param})$ with $\mathsf{pk} = \big(\mathbf{G}, \mathbf{Q}, \mathbf{s} = \mathsf{sk}\cdot \mathbf{h}^\top\big)$
2b. $\mathbf{s} \xleftarrow{\$} \mathcal{R}$
2c. $(\mathsf{pk}, \mathsf{sk}) \leftarrow ((\mathbf{G}, \mathbf{Q}, \mathbf{s}), \mathbf{0})$
3. $(\mathbf{m}_0, \mathbf{m}_1) \leftarrow \mathcal{A}(\mathtt{FIND} : \mathsf{pk})$
4a. Use randomness $\theta$ to generate $\mathbf{e}' \xleftarrow{\$} \mathcal{R}$, $\mathbf{r} = (\mathbf{r}'_1, \mathbf{r}'_2) \xleftarrow{\$} \mathcal{R}^2$ uniformly at random
```

```
4b. $\mathbf{u}^\top \leftarrow \mathbf{Q}\mathbf{r'}^\top$ and $\mathbf{v} \leftarrow \mathbf{m}_1\mathbf{G} + \mathbf{s} \cdot \mathbf{r'}_2 + \mathbf{e'}$
4c. $\mathbf{c}^* \leftarrow (\mathbf{u}, \mathbf{v})$
5. $\mathbf{b'} \leftarrow \mathcal{A}(\texttt{GUESS} : \mathbf{c}^*)$
6. RETURN $\mathbf{b'}$
```

The outputs from Game $\mathbf{G}_3$ and Game $\mathbf{G}_4$ follow the exact same distribution, and therefore the two games are indistinguishable from an information-theoretic point of view. Indeed, for each tuple $(\boldsymbol{r}, \mathbf{e})$ of Game $\mathbf{G}_3$, resulting in a given $(\mathbf{u}, \mathbf{v})$, there is a one to one mapping to a couple $(\boldsymbol{r}', \mathbf{e}')$ resulting in Game $\mathbf{G}_4$ in the *same* $(\mathbf{u}, \mathbf{v})$, namely $\boldsymbol{r}' = \boldsymbol{r}$ and $\mathbf{e}' - \mathbf{m}_0\mathbf{G} + \mathbf{m}_1\mathbf{G}$. This implies that choosing uniformly $(\boldsymbol{r}, \mathbf{e})$ in Game $\mathbf{G}_3$ and choosing uniformly $(\boldsymbol{r}', \mathbf{e}')$ in Game $\mathbf{G}_4$ leads to the same output distribution for $(\mathbf{u}, \mathbf{v})$.

Game $\mathbf{G}_5$: In this game, we now pick $\mathbf{r}'_1, \mathbf{r}'_2, \mathbf{e}'$ with the correct weight.

Game $\mathbf{G}_6$: We now conclude by switching the public key to an honestly generated one.

We do not explicit these last two games as Game $\mathbf{G}_4$ and Game $\mathbf{G}_5$ are the equivalents of Game $\mathbf{G}_3$ and Game $\mathbf{G}_2$ except that $\mathbf{m}_1$ is used instead of $\mathbf{m}_0$. A distinguisher between these two games breaks therefore the 3-DRQCSD assumption too. Similarly Game $\mathbf{G}_5$ and Game $\mathbf{G}_6$ are the equivalents of Game $\mathbf{G}_2$ and Game $\mathbf{G}_1$ and a distinguisher between these two games breaks the 2-DRQCSD assumption.

We managed to build a sequence of games allowing a simulator to transform a ciphertext of a message $\mathbf{m}_0$ to a ciphertext of a message $\mathbf{m}_1$. Hence, the advantage of an adversary against the IND-CPA experiment is bounded as:

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{E},\mathcal{A}}(\lambda) \leq 2\left(\mathsf{Adv}^{\mathsf{2\text{-}DRQCSD}}(\lambda) + \mathsf{Adv}^{\mathsf{3\text{-}DRQCSD}}(\lambda)\right). \tag{18}$$

□

5 Known Attacks

There exist two types of generic attacks on these problems:

- the combinatorial attacks where the goal is to find the support of the error or of the codeword.
- the algebraic attacks where the opponent tries to solve an algebraic system by Groebner basis.

First, we deal with the combinatorial attacks then we discuss the algebraic attacks.

5.1 Generic attacks

For $\mathcal{C}$ a $[n, k]$ rank code over $\mathbb{F}_{q^m}$, the best combinatorial attacks to decode a word with an error of weight r is:

$$\mathcal{O}\left((nm)^3 q^{r\left\lceil \frac{m(k+1)}{n} \right\rceil - m}\right)$$

This attack is an improvement of a previous attack described in [12], a detailed description of the attack can be found in [3]. The general idea of the attack is to adapt the Information Set Decoding attack for Hamming distance to rank metric. For rank metric the attacker tries to guess a subspace which contains the support of the error and test whether the choice of the subspace contains the support of the error or not, by solving a system of syndrome equations. There is no known attack which uses the quasi-cyclicity of a code to improve upon this attack, whenever the polynomial $X^N - 1 \mod q$ has no small factors except $X - 1$ [16].

5.2 Algebraic attacks

The second way to solve the equations of the system defined by the RSD problem is to use Groebner basis [19]. The advantage of these attacks is that they are independent of the size of q. They mainly depend on the number of unknowns with respect to the number of equations. However, in the case $q = 2$ the number of unknowns is generally too high for that the algorithms by Groebner basis are more efficient than the combinatorial attacks. We have chosen our parameters such that the best attacks are combinatorial, the expected complexity of the algorithms by Groebner basis is based on the article [5].

6 Advantages and Limitations

6.1 Advantages

The main advantages of RQC over existing code-based cryptosystems are:

- its IND-CPA reduction to a well-understood problem on coding theory: the Syndrome Decoding problem,
- its immunity against attacks aiming at recovering the hidden structure of the code being used,
- it features a null decryption failure rate.

The last item allows to achieve a tight reduction for the IND-CCA2 security of the KEM-DEM version through the recent transformation of [17].

6.2 Limitations

The objects considered (codes over extension fields) may seem hard to manipulate , but in practice the results obtained show good execution times.

References

[1] Carlos Aguilar Melchor, Olivier Blazy, Jean Christophe Deneuville, Philippe Gaborit, and Gilles Zémor. Efficient encryption from random quasi-cyclic codes. *CoRR*, abs/1612.05572, 2016. `http://arxiv.org/abs/1612.05572`. *3*

[2] 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, Heidelberg, August 2007. *10*

[3] Nicolas Aragon, Philippe Gaborit, Adrien Hauteville, and Jean-Pierre Tillich. Improvement of generic attacks on the rank syndrome decoding problem., 2017. Pre-print, available at `https://www.unilim.fr/pages_perso/philippe.gaborit/newGRS.pdf`. *15*, *23*

[4] Daniel Augot, Pierre Loidreau, and Gwezheneg Robert. Generalized gabidulin codes over fields of any characteristic. *arXiv preprint arXiv:1703.09125*, 2017. *8*, *9*

[5] Luk Bettale, Jean-Charles Faugere, and Ludovic Perret. Hybrid approach for solving multivariate systems over finite fields. *Journal of Mathematical Cryptology*, 3(3):177–197, 2009. *23*

[6] Jean-Christophe Deneuville, Philippe Gaborit, and Gilles Zémor. Ouroboros: A simple, secure and efficient key exchange protocol based on coding theory. In Tanja Lange and Tsuyoshi Takagi, editors, *Post-Quantum Cryptography - 8th International Workshop, PQCrypto 2017, Utrecht, The Netherlands, June 26-28, 2017, Proceedings*, volume 10346 of *Lecture Notes in Computer Science*, pages 18–34. Springer, 2017. `http://www.unilim.fr/pages_perso/deneuville/files/ba43bf8d80cef2999dbf4308828213ec.pdf`. *3*

[7] Ernest Mukhamedovich Gabidulin. Theory of codes with maximum rank distance. *Problemy Peredachi Informatsii*, 21(1):3–16, 1985. *6*, *7*

[8] Ernst M. Gabidulin, A. V. Paramonov, and O. V. Tretjakov. Ideals over a non-commutative ring and thier applications in cryptology. In Donald W. Davies, editor, *EUROCRYPT'91*, volume 547 of *LNCS*, pages 482–489. Springer, Heidelberg, April 1991. `http://link.springer.com/chapter/10.1007/3-540-46416-6_41`. *7*

[9] Philippe Gaborit. Shorter keys for code based cryptography. In *Proceedings of the 2005 International Workshop on Coding and Cryptography (WCC 2005)*, pages 81–91, 2005. `http://www.unilim.fr/pages_perso/philippe.gaborit/shortIC.ps`. *4*

[10] Philippe Gaborit and Marc Girault. Lightweight code-based identification and signature. In *2007 IEEE International Symposium on Information Theory*, pages 191–195. IEEE, 2007. `https://www.unilim.fr/pages_perso/philippe.gaborit/isit_short_rev.pdf`. *10*

[11] Philippe Gaborit, Adrien Hauteville, and Jean-Pierre Tillich. Ranksynd a PRNG based on rank metric. In Tsuyoshi Takagi, editor, *Post-Quantum Cryptography - 7th International Workshop, PQCrypto 2016, Fukuoka, Japan, February 24-26, 2016, Proceedings*, volume 9606 of *Lecture Notes in Computer Science*, pages 18–28. Springer, 2016. `https://arxiv.org/pdf/1603.05128.pdf`. *15*

[12] Philippe Gaborit, Olivier Ruatta, and Julien Schrek. On the complexity of the rank syndrome decoding problem. *IEEE Transactions on Information Theory*, 62(2):1006–1019, 2016. `https://arxiv.org/pdf/1301.1026.pdf`. *15*, *23*

[13] 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. `https://arxiv.org/pdf/1404.3482.pdf`. *9*

[14] Pierrick Gaudry and Emmanuel Thomé. The mpfq library and implementing curve-based key exchanges. In *SPEED: software performance enhancement for encryption and decryption*, pages 49–64, 2007. *17*

[15] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984. *11*

[16] Adrien Hauteville and Jean-Pierre Tillich. New algorithms for decoding in the rank metric and an attack on the lrpc cryptosystem. In *2015 IEEE International Symposium on Information Theory (ISIT)*, pages 2747–2751. IEEE, 2015. `https://arxiv.org/pdf/1504.05431.pdf`. *23*

[17] Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. A modular analysis of the fujisaki-okamoto transformation. Cryptology ePrint Archive, Report 2017/604, 2017. `http://eprint.iacr.org/2017/604`. *3*, *12*, *14*, *18*, *23*

[18] W Cary Huffman and Vera Pless. *Fundamentals of error-correcting codes*. Cambridge university press, 2010. `https://www.amazon.fr/Fundamentals-Error-Correcting-Codes-Cary-Huffman/dp/0521131707`. *4*

[19] Françoise Levy-dit Vehel and L Perret. Algebraic decoding of rank metric codes. *Proceedings of YACC*, 2006. *23*

[20] Pierre Loidreau. Properties of codes in rank metric. *arXiv preprint cs/0610057*, 2006. `https://arxiv.org/pdf/cs/0610057.pdf`. *5*, *6*

[21] Pierre Loidreau. A welch–berlekamp like algorithm for decoding gabidulin codes. In *Coding and cryptography*, pages 36–45. Springer, 2006. *8*, *9*

[22] Rafael Misoczki, Jean-Pierre Tillich, Nicolas Sendrier, and Paulo SLM Barreto. Mdpc-mceliece: New mceliece variants from moderate density parity-check codes. In *Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on*, pages 2069–2073. IEEE, 2013. `https://eprint.iacr.org/2012/409.pdf`. *5*, *10*

[23] Oystein Ore. On a special class of polynomials. *Transactions of the American Mathematical Society*, 35(3):559–584, 1933. *7*

[24] Raphael Overbeck. A new structural attack for GPT and variants. In *Mycrypt*, volume 3715, pages 50–63, 2005. `http://link.springer.com/chapter/10.1007/11554868_5`. *7*

[25] Irving S Reed and Gustave Solomon. Polynomial codes over certain finite fields. *Journal of the society for industrial and applied mathematics*, 8(2):300–304, 1960. *7*

[26] Nicolas Sendrier. Decoding one out of many. In *International Workshop on Post-Quantum Cryptography*, pages 51–67. Springer, 2011. `https://eprint.iacr.org/2011/367.pdf`. *10*

[27] Danilo Silva, Frank R Kschischang, and Ralf Kotter. Communication over finite-field matrix channels. *IEEE Transactions on Information Theory*, 56(3):1296–1305, 2010. *7*