Rank Quasi-Cyclic (RQC)

Second Round version - update for April 21st, 2020

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. The main features of the RQC submission are:

- IND-CCA2 KEM ;
- Small public key size ;
- Null decryption failure rate ;
- Efficient implementations based on classical decoding algorithms.

Principal Submitters (by alphabetical order):

- Carlos Aguilar Melchor
- Nicolas Aragon
- Slim Bettaieb
- Loïc Bidoux
- Olivier Blazy
- Maxime Bros
- Alain Couvreur
- Jean-Christophe Deneuville
- Philippe Gaborit
- Adrien Hauteville
- Gilles Zémor

Inventors: Same as submitters

Developers: Same as submitters

Owners: Same as submitters

(The PhD of N. Aragon is partially funded by French DGA.)

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

@ jean-christophe.deneuville@enac.fr

+33-631-142-705

ENAC Toulouse

7 avenue Edouard Belin
31 400 Toulouse
France

1 History of updates on RQC

1.1 Updates for April 21st, 2020

- New results on rank metric [5, 6] now provide a clear and better understanding of algebraic attacks against the rank syndrome decoding problem. As a consequence, we have introduced a minor modification to the protocol. In the encryption, $\mathbf{r}_1$, $\mathbf{r}_2$ and $\mathbf{e}$ no longer share the same support but instead $\mathbf{r}_1$ and $\mathbf{r}_2$ share a support of size w_1 and $\mathbf{e}$ has a support of size $w_1 + w_2$ such that $\text{Supp}(\mathbf{r}_1, \mathbf{r}_2) \subset \text{Supp}(\mathbf{e})$.

  This minor modification enables us to keep low parameters for the scheme even if the rank weights of the instances have been increased. It leads us to introduce a new problem where the support of the error is not homogeneous: the Non Homogeneous IRSD problem (Section 2.1.4). This problem is a small variation on the IRSD problem, and it has a straightforward reduction from the IRSD problem for rank weight w_1. For combinatorial attacks, the increase of parameters permits to still have a sufficient reduction to the IRSD problem for rank weight w_1. For algebraic attacks, we consider a specialization of the recent best known attacks in this case (Section 6, from [6]).

- We have updated parameters for our scheme (increase of order of 40% for RQC-I-128 compared to Round 2 submission, see Table 3).

- We provide an optimized implementation leveraging AVX and CLMUL instructions.

- Our implementations are now implemented in a constant-time way whenever relevant and they should not leak any sensitive information with respect to timing attacks.

- Our implementations no longer rely on third party libraries for finite field arithmetic.

- We welcome Maxime Bros as a new member of our team.

- For 128 bits of security, we obtain the following sizes (in bytes), performances (in kilocycles) and Decryption Failure Rate for RQC:

Public Key	Ciphertext	KeyGen	Encaps	Decaps	DFR
1,874	3,652	370	530	2580	0

1.2 Updates between Round 1 and Round 2

- RQC now uses ideal codes instead of quasi-cyclic codes.

- The parameters of the scheme have been updated so that the weight of the error, which is the most important parameter for the security, increases regularly with each level of security. In practice, it leads to a small increase of the parameters.

- The supporting documentation has been reorganized for clarification. Besides, additional details on Gabidulin codes, quantum speed-up on known attacks and resistance to timing attacks have been added.

- The reference implementation have been improved in several ways. In particular, finite field arithmetic now relies on NTL rather than MPFQ.

- Two additional members have been added to the RQC proposal: Alain COUVREUR and Adrien HAUTEVILLE.

Contents

2 Specifications

In this section, we introduce RQC, an encryption scheme based on coding theory. RQC stands for Rank Quasi-Cyclic. This proposal has been published in IEEE Transactions on Information Theory [1]. RQC is a code-based public key cryptosystem with several desirable properties:

- It is proven IND-CPA assuming the hardness of (a decisional version of) the Rank Syndrome Decoding problem on structured codes. By construction, RQC perfectly fits the recent KEM-DEM transformation of [15], and allows to get an hybrid encryption scheme with strong security guarantees (IND-CCA2).
- 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.
- The decryption algorithm is deterministic so the Decryption Failure Rate (DFR) is zero.
- 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. 2.1, we make some recalls on encryption and security in Sec. 2.1.5 then present our proposal in Sec. 2.2. Concrete sets of parameters are provided in Sec. 2.4.

2.1 Preliminaries

2.1.1 General definitions

In the following document, 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 (resp. uppercase) letters to denote vectors (resp. matrices).

Let $P \in \mathbb{F}_q[X]$ a polynomial of degree n. We can identify the vector space $\mathbb{F}_{q^m}^n$ with the ring $\mathbb{F}_{q^m}[X]/\langle P \rangle$, where $\langle P \rangle$ denotes the ideal of $\mathbb{F}_{q^m}[X]$ generated by P.

$$\begin{aligned}\Psi : \quad \mathbb{F}_{q^m}^n &\simeq \mathbb{F}_{q^m}[X]/\langle P \rangle \\ (v_0, \ldots, v_{n-1}) &\mapsto \sum_{i=0}^{n-1} v_i X^i\end{aligned}$$

For $\mathbf{u}, \mathbf{v} \in \mathbb{F}_{q^m}^n$, we define their product similarly as in $\mathbb{F}_{q^m}[X]/\langle P \rangle$: $\mathbf{w} = \mathbf{uv} \in \mathbb{F}_{q^m}^n$ is the only vector such that $\Psi(\mathbf{w}) = \Psi(\mathbf{u})\Psi(\mathbf{v})$. In order to lighten the formula, we will omit the symbol Ψ in the future.

To a vector $\mathbf{v} \in \mathbb{F}_{q^m}^n$ we can associate an $n \times n$ square matrix with entries in $\mathbb{F}_{q^m}^n$ corresponding to the product by $\mathbf{v}$. Indeed,

$$\begin{aligned}
\mathbf{u} \cdot \mathbf{v} &= \mathbf{u}(X)\mathbf{v}(X) \pmod{P} \\
&= \sum_{i=0}^{n-1} u_i X^i \mathbf{v}(X) \pmod{P} \\
&= \sum_{i=0}^{n-1} u_i (X^i \mathbf{v}(X) \mod P) \\
&= (u_0, \ldots, u_{n-1}) \begin{pmatrix} \mathbf{v}(X) \mod P \\ X\mathbf{v}(X) \mod P \\ \vdots \\ X^{n-1}\mathbf{v}(X) \mod P \end{pmatrix}.
\end{aligned}$$

Such a matrix is called the ideal matrix generated by $\mathbf{v}$ and P, or simply by $\mathbf{v}$ when there is no ambiguity in the choice of P.

Definition 2.1.1 (Ideal Matrix). *Let $P \in \mathbb{F}_q[X]$ be a polynomial of degree n and $\mathbf{v} \in \mathbb{F}_{q^m}^n$. The ideal matrix generated by $\mathbf{v}$ is the $n \times n$ square matrix denoted $\mathcal{IM}(\mathbf{v})$ of the form:*

$$\mathcal{IM}(\mathbf{v}) = \begin{pmatrix} \mathbf{v} \\ X\mathbf{v} \mod P \\ \vdots \\ X^{n-1}\mathbf{v} \mod P \end{pmatrix}.$$

As a consequence, the product of two elements of $\mathbb{F}_{q^m}[X]/\langle P \rangle$ is equivalent to the usual vector-matrix product:

$$\mathbf{u} \cdot \mathbf{v} = \mathbf{u}\mathcal{IM}(\mathbf{v}) = \mathcal{IM}(\mathbf{u})^T \mathbf{v} = \mathbf{v} \cdot \mathbf{u}.$$

Definition 2.1.2 (Rank metric over $\mathbb{F}_{q^m}^n$). *Let $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{F}_{q^m}^n$ and $(\beta_1, \ldots, \beta_m) \in \mathbb{F}_{q^m}^m$ be a basis of $\mathbb{F}_{q^m}$ viewed as an m-dimensional vector space over $\mathbb{F}_q$. Each coordinate x_j is associated to a vector of $\mathbb{F}_q^m$ in this basis: $x_j = \sum_{i=1}^m x_{ij}\beta_i$. The $m \times n$ matrix associated to $\mathbf{x}$ is given by $\mathbf{M}(\mathbf{x}) = (x_{ij})_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}}$.*

The rank weight $\|\mathbf{x}\|$ of $\mathbf{x}$ is defined as

$$\|\mathbf{x}\| \overset{def}{=} \operatorname{Rank} \mathbf{M}(\mathbf{x}).$$

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

Definition 2.1.3 ($\mathbb{F}_{q^m}$-linear code). *An $\mathbb{F}_{q^m}$-linear code $\mathcal{C}$ of dimension k and length n is a subspace of dimension k of $\mathbb{F}_{q^m}^n$ embedded with the rank metric. It is denoted $[n, k]_{q^m}$.*

Such a code $\mathcal{C}$ can be represented in two equivalent ways:

- *by a generator matrix* $\mathbf{G} \in \mathbb{F}_{q^m}^{k\times n}$. *Each row of* $\mathbf{G}$ *is an element of a basis of* $\mathcal{C}$,

$$\mathcal{C} = \{\mathbf{xG}, \mathbf{x} \in \mathbb{F}_{q^m}^{k}\}.$$

- *by a parity-check matrix* $\mathbf{H} \in \mathbb{F}_{q^m}^{(n-k)\times n}$. *Each row of* $\mathbf{H}$ *determines a parity-check equation verified by the elements of* $\mathcal{C}$:

$$\mathcal{C} = \{\mathbf{x} \in \mathbb{F}_{q^m}^{n} : \mathbf{H}\mathbf{x}^T = \mathbf{0}\}.$$

  $\mathbf{H}\mathbf{v}^T$ *is called the* syndrome *of* $\mathbf{v}$ *(with respect to* $\mathbf{H}$*).*

We say that $\mathbf{G}$ *(respectively* $\mathbf{H}$*) is under systematic form if and only if it is of the form* $(\mathbf{I}_k|\mathbf{A})$ *(respectively* $(\mathbf{I}_{n-k}|\mathbf{B})$*).*

Definition 2.1.4 (Support of a word). *Let* $\mathbf{x} = (x_1, \dots, x_n) \in \mathbb{F}_{q^m}^n$. *The support* E *of* $\mathbf{x}$, *denoted* $\mathrm{Supp}(\mathbf{x})$, *is the* $\mathbb{F}_q$*-subspace of* $\mathbb{F}_{q^m}$ *generated by the coordinates of* $\mathbf{x}$:

$$E = \langle x_1, \dots, x_n \rangle_{\mathbb{F}_q}$$

and we have $\dim(E) = \|\mathbf{x}\|$.

The number of supports of dimension w of $\mathbb{F}_{q^m}$ is given by the Gaussian coefficient

$$\begin{bmatrix} m \\ w \end{bmatrix}_q = \prod_{i=0}^{w-1} \frac{q^m - q^i}{q^w - q^i}.$$

2.1.2 Ideal codes

One of the drawbacks of code-based cryptography is the size of the keys. Indeed, to represent an $[n,k]_{q^m}$ code with a systematic matrix, we need $k(n-k)$ symbols in $\mathbb{F}_{q^m}$, or $k(n-k)m\lceil \log q \rceil$ bits. In order to reduce the size of the representation of a code, we introduce the family of ideal codes, which are basically codes with a systematic generator matrix formed with blocks of ideal matrices. More formally,

Definition 2.1.5 (Ideal codes). *Let* $P(X) \in \mathbb{F}_q[X]$ *be a polynomial of degree* n. *An* $[ns, nt]_{q^m}$ *code* $\mathcal{C}$ *is an* (s,t)*-ideal code if its generator matrix under systematic form is of the form*

$$\mathbf{G} = \left(\begin{array}{c|ccc} & \mathcal{IM}(\mathbf{g}_{1,1}) & \dots & \mathcal{IM}(\mathbf{g}_{1,s-t}) \\ \mathbf{I}_{tn} & \vdots & \ddots & \vdots \\ & \mathcal{IM}(\mathbf{g}_{t,1}) & \dots & \mathcal{IM}(\mathbf{g}_{t,s-t}) \end{array} \right)$$

where $(\mathbf{g}_{i,j})_{\substack{i\in[1..s-t] \\ j\in[1..t]}}$ *are vectors of* $\mathbb{F}_{q^m}^n$. *In this case, we say that* $\mathcal{C}$ *is generated by the* $(\mathbf{g}_{i,j})$.

It would be somewhat more natural to choose the generator matrix to be made up of $s \times t$ ideal matrices, rather than to require the code to admit a systematic generator matrix. However, if m and n are two different prime numbers and if P is irreducible, a non-zero ideal matrix is always non-singular. To prove this, we need the following lemma:

Lemma 1. *Let m and n be two different prime numbers. Let $P \in \mathbb{F}_q[X]$ be an irreducible polynomial of degree n and $U \in \mathbb{F}_{q^m}[X]$ a non zero polynomial of degree at most $n-1$. Then P and U are co-prime in $\mathbb{F}_{q^m}[X]$.*

Proof. We will show that P and U have no common root. Let $\mathcal{Z}(P)$ (respectively $\mathcal{Z}(U)$) be the set of the roots of P (respectively U) in an algebraic closure of $\mathbb{F}_q$.

Since P is irreducible of degree n, its roots generate $\mathbb{F}_{q^n}$

$$\implies \mathcal{Z}(P) \subset \mathbb{F}_{q^n} \backslash \mathbb{F}_q$$

Since U is of degree at most $n-1$, its roots belong to $\mathbb{F}_{q^{m(n-1)!}}$.

But $\text{GCD}(n, m(n-1)!) = 1$ when m and n are two different prime numbers. Thus

$$\mathbb{F}_{q^{m(n-1)!}} \cap \mathbb{F}_{q^n} = \mathbb{F}_q \implies \mathcal{Z}(P) \cap \mathcal{Z}(U) = \emptyset$$

Hence, P and U are co-prime. □

Now, let $\mathbf{u} \in \mathbb{F}_{q^m}^n$ a non zero vector and $P \in \mathbb{F}_q[X]$ an irreducible polynomial of degree n. According to the previous lemma, there exists a vector $\mathbf{v} \in \mathbb{F}_{q^m}^n$ such that

$$\begin{aligned} &\mathbf{uv} = 1 \pmod{P} \\ \iff &\mathbf{u}\mathcal{IM}(\mathbf{v}) = (1, 0, \ldots, 0) \\ \iff &\mathcal{IM}(\mathbf{u})\mathcal{IM}(\mathbf{v}) = \mathbf{I}_n \end{aligned}$$

This demonstrates that every block of ideal matrix of $\mathbf{G}$ is non-singular, hence $\mathcal{C}$ can be represented under systematic form.

All the parameters we propose in Section 2.4 verify these conditions.

Remark 2.1. *With this definition, ideal codes can be seen as a generalization of Quasi-Cyclic codes. Indeed, the generator matrix under systematic form of a Quasi-Cyclic code [9] is of the same form, except that the ideal matrices are replaced by circulant matrices. Yet, an $n \times n$ circulant matrix can be seen as an element of $\mathbb{F}_{q^m}[X]/\langle X^n - 1\rangle$. Thus ideal codes only differ from Quasi-Cyclic codes by the choice of the polynomial P.*

In our scheme, we only use $[ns, n]_{q^m}$ ideal codes. In order to shorten the notation, we denote these codes an s-ideal code. If $\mathcal{C}$ is an $[sn, n]$ ideal code generated by $(\mathbf{g}_1, \ldots, \mathbf{g}_{s-1})$, we have $\mathcal{C} = \{(\mathbf{u}, \mathbf{ug}_1, \ldots, \mathbf{ug}_{s-1}), \mathbf{u} \in \mathbb{F}_{q^m}^n\}$.

We need to be careful when we use this notation in the case of parity-check matrix. Indeed, the parity-check matrix under systematic form of $\mathcal{C}$ is of the form:

$$\mathbf{H} = \begin{pmatrix} & \mathcal{IM}(\mathbf{h}_1)^T \\ \mathbf{I}_{n(s-1)} & \vdots \\ & \mathcal{IM}(\mathbf{h}_{s-1})^T \end{pmatrix}. \tag{1}$$

Thus, if $\boldsymbol{\sigma} = (\boldsymbol{\sigma}_1 \dots \boldsymbol{\sigma}_{s-1}) \in \mathbb{F}_{q^m}^{s(n-1)}$ is the syndrome of an error $\mathbf{e} = (\mathbf{e}_1 \dots \mathbf{e}_{s-1}) \in \mathbb{F}_{q^m}^{ns}$, the parity-check equations

$$\mathbf{H}\mathbf{e}^T = \boldsymbol{\sigma}^T$$

are equivalent to $\mathbf{e}_i + \mathbf{h}_i\mathbf{e}_{s-1} = \boldsymbol{\sigma}_i$ for $1 \leqslant i \leqslant s-1$.

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 [17] as

$$r \leq 1 + \left\lfloor \frac{(n-k)m}{n} \right\rfloor . \tag{2}$$

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

Deterministic Decoding. Unlike with 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 weight. 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 [8]. More details about these codes are provided in the next section. 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 [22] 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.

2.1.3 Gabidulin codes and their decoding

Gabidulin codes were introduced in 1985 [8]. These codes are analog to Reed-Solomon codes in Hamming metric [21], but involve q-polynomials instead of regular ones. They have therefore a strong algebraic structure. The notion of q-polynomials was introduced by Ore [19], we hereafter give some background.

Definition 2.1.6 (q-polynomials)**.** *The set of q-polynomials over $\mathbb{F}_{q^m}$ is the set of polynomials with the following shape:*

$$\left\{ P(X) = \sum_{i=0}^{r} p_i X^{q^i}, \ \text{with } p_i \in \mathbb{F}_{q^m} \text{ and } p_r \neq 0 \right\}.$$

The q-degree of a q-polynomial P is defined as $\deg_q(P) = r$.

Definition 2.1.7 (Ring of q-polynomials)**.** *The set of q-polynomials over $\mathbb{F}_{q^m}$ is a non-commutative ring when considered with the following operations:*

- *Addition:* $(P+Q)(X) = P(X) + Q(X)$
- *Composition:* $(P \circ Q)(X) = P[Q(X)]$

Due to their structure, the q-polynomials are inherently related to decoding problems in the rank metric as stated by the following propositions.

Theorem 2.2 ([19])**.** *Any $\mathbb{F}_q$-subspace of $\mathbb{F}_{q^m}$ of dimension r is the set of the roots of a unique unitary q-polynomial P such that $\deg_q(P) = r$.*

Corollary 2.1. *Let $\mathbf{x} = (x_1, x_2, \dots, x_n) \in \mathbb{F}_{q^m}^n$ and V be the unitary q-polynomial of smallest q-degree such that $V(x_i) = 0$ for $1 \leq i \leq n$, then $\|\mathbf{x}\| = r$ if and only if $\deg_q(V) = r$.*

Gabidulin codes can be thought as the evaluation of q-polynomials of bounded degree on the coordinates of a vector over $\mathbb{F}_{q^m}$.

Definition 2.1.8 (Gabidulin codes)**.** *Let $k, n, m \in \mathbb{N}$ such that $k \leqslant n \leqslant m$. Let $\mathbf{g} = (g_1, \dots, g_n)$ be a $\mathbb{F}_q$ linearly independent family of elements of $\mathbb{F}_{q^m}$. The* Gabidulin code *$\mathcal{G}_{\mathbf{g}}(n, k, m)$ is the following code $[n, k]_{q^m}$:*

$$\left\{ P(\mathbf{g}), \deg_q P < k \right\} \text{ where } P(\mathbf{g}) := (P(g_1), \dots, P(g_v)).$$

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 benefit from an efficient decoding algorithm correcting up to $\left\lfloor \frac{n-k}{2} \right\rfloor$ errors in a deterministic way [8].

Decoding Gabidulin codes. The algorithm employed in order to decode Gabidulin codes has been proposed in [18] 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 2.1.9 (**Decoding**$(\mathbf{y}, \mathcal{G}_{\mathbf{g}}, t)$ [18])**.** *Find, if it exists, $\mathbf{c} \in \mathcal{G}_{\mathbf{g}}$ and $\mathbf{e}$ with $\|\mathbf{e}\| \leq t$ such that $\mathbf{y} = \mathbf{c} + \mathbf{e}$.*

One can decode Gabidulin codes using the q-polynomial reconstruction problem:

Definition 2.1.10 (**Reconstruction**$(\mathbf{y}, \mathbf{g}, k, t)$ [18])**.** *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 2.3 ([18])**.** *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 2.1.11 (**Reconstruction2**$(\mathbf{y}, \mathbf{g}, k, t)$ [18])**.** *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 2.4 ([18])**.** *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 quotient of* N *by* V *by left euclidean division in the ring of* q*-polynomials is a solution of* **Reconstruction**$(\mathbf{y}, \mathbf{g}, k, t)$.

The algorithm described in [4] (see Section 4, Algorithm 5) can be used in order to solve the **Reconstruction2**$(\mathbf{y}, \mathbf{g}, k, t)$ problem. This algorithm can be decomposed in two steps:

1. *Initialization step*: Two pairs of q-polynomials (N_0, V_0) and (N_1, V_1) satisfying $V_0(y_i) = N_0(g_i)$ and $V_1(y_i) = N_1(g_i)$ for $1 \leq i \leq k$ are computed. To do so, N_0 is defined as the annihilator q-polynomial on $g_1, \ldots, g_k$, N_1 is defined as the q-polynomial interpolating $y_1, \ldots, y_k$ on $g_1, \ldots, g_k$ while $V_0(X) = 0$ and $V_1(X) = X$.

2. *Interpolation step*: Iteratively, the q-degrees of (N_0, V_0) and (N_1, V_1) are increased using a recurrence relation ensuring that if the interpolation conditions $V(y_i) = N(g_i)$ for $1 \leq i \leq j$ are satisfied at step j, then they are also satisfied at step $j+1$. Besides, this construction ensures that at least one of the pairs satisfies the final degree conditions $\deg_q(V) \leq t$ and $\deg_q(N) \leq k + t - 1$ when the q-polynomials are initialized according to the aforementioned Initialization step.

The correctness of this algorithm is provided in [4], Theorem 10. Using a solution of **Reconstruction2**$(\mathbf{y}, \mathbf{g}, k, t)$, one can solve **Decoding**$(\mathbf{y}, \mathcal{G}_{\mathbf{g}}, t)$ for Gabidulin codes as follows:

Definition 2.1.12 (**Algorithm for Decoding**$(\mathbf{y}, \mathcal{G}_{\mathbf{g}}, t)$ [18, 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 2.5 ([4])**.** *The complexity of solving* **Decoding**$(\mathbf{y}, \mathcal{G}_{\mathbf{g}}, t)$ *by using the algorithm described in Definition 2.1.12 is* $\mathcal{O}(n^2)$ *operations in* $\mathbb{F}_{q^m}$*.*

Note: Gabidulin decoding has been implemented using the "Polynomials with lower degree" optimization; see Section 4.4.2 of [4] for additional details.

2.1.4 Difficult problems for cryptography

In this section, we describe difficult problems which can be used for cryptography and discuss their hardness. 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 2.1.13 (RSD Distribution)**.** *For positive integers,* n*,* k*, and* w*, the* $\mathsf{RSD}(n, k, w)$ Distribution *chooses* $\mathbf{H} \overset{\$}{\leftarrow} \mathbb{F}_{q^m}^{(n-k)\times n}$ *and* $\mathbf{x} \overset{\$}{\leftarrow} \mathbb{F}_{q^m}^{n}$ *such that* $\|\mathbf{x}\| = w$*, and outputs* $(\mathbf{H}, \sigma(\mathbf{x}) = \mathbf{H}\mathbf{x}^\top)$*.*

Definition 2.1.14 (Computational 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* Computational 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* $\|\mathbf{x}\| = w$*.*

The RSD problem has recently been proven difficult with a probabilistic reduction from the Hamming setting in [12]. For cryptography we also need a decision version of the problem, which is given in the following definition.

Definition 2.1.15 (Decisional RSD Problem)**.** *On input* $(\mathbf{H}, \mathbf{y}^\top) \in \mathbb{F}_{q^m}^{(n-k)\times n} \times \mathbb{F}_{q^m}^{(n-k)}$*, the* Decisional 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 uses ideal codes, we explicitly define the problem for this setting. The following definitions describe the DRSD problem in the ideal configuration, and are just a combination of Definition 2.1.5 and 2.1.15. Ideal codes are very useful in cryptography since their compact description allows to decrease considerably the size of the keys.

Definition 2.1.16 (s-IRSD Distribution)**.** *For positive integers n, w, and s, let $P \in \mathbb{F}_q[X]$ be an irreducible polynomial of degree n, let $S(n,s)$ be the set of the parity-check matrices $\mathbf{H}$ under systematic form of s-ideal codes of type $[sn,n]$ (see Equation 1). The s-$\mathsf{IRSD}(n,w)$* Distribution *chooses uniformly at random a matrix $\mathbf{H} \overset{\$}{\leftarrow} S(n,s)$ together with a vector $\mathbf{x} = (\mathbf{x}_1,\ldots,\mathbf{x}_s) \overset{\$}{\leftarrow} \mathbb{F}_{q^m}^{sn}$ such that $\|\mathbf{x}\| = w$ and outputs $(\mathbf{H}, \mathbf{H}\mathbf{x}^\top)$.*

Definition 2.1.17 (Computational s-IRSD Problem)**.** *For positive integers n, w, and s, let $P \in \mathbb{F}_q[X]$ be an irreducible polynomial of degree n, let $\mathbf{H}$ be a random parity check matrix under systematic form of an s-ideal code $\mathcal{C}$ and $\mathbf{y} \overset{\$}{\leftarrow} \mathbb{F}_{q^m}^{sn-n}$, the* Computational s-ideal RSD Problem s-$\mathsf{IRSD}(n,w)$ *asks to find $\mathbf{x} = (\mathbf{x}_1,\ldots,\mathbf{x}_s) \in \mathbb{F}_{q^m}^{sn}$ such that $\|\mathbf{x}\| = w$ and $\mathbf{y} = \mathbf{x}\mathbf{H}^\top$.*

Assumption 1. *Although there is no general complexity result for ideal codes, decoding these codes is considered as hard by the community. For the rank metric, there is no known generic attack which exploits the ideal structure of the code. Thus, in practice, the best attacks are the same as those for arbitrary codes.*

The problem has a decisional form:

Definition 2.1.18 (Decisional s-IRSD Problem)**.** *For positive integers n, w, and s, let $P \in \mathbb{F}_q[X]$ be an irreducible polynomial of degree n, let $S(n,s)$ be the set of the parity-check matrices $\mathbf{H}$ under systematic form of s-ideal codes of type $[sn,n]$ (see Equation 1). The* Decisional s-Ideal RSD Problem s-$\mathsf{DIRSD}(n,w)$ *asks to decide with non-negligible advantage whether $(\mathbf{H}, \mathbf{y}^\top)$ came from the s-IRSD(n,w) distribution or the uniform distribution over $S(n,s) \times \mathbb{F}_{q^m}^{(sn-n)}$.*

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

In what follows, we will need a version of the s-IRSD Problem in which the error has a *non-homogeneous* (NH) weight. This means that some coordinates of the error will have a support E_1 whereas the others will have a different support E_2. It is easy to give a general definition of this new problem, nevertheless for the sake of clarity, we will just define the specific one we need.

Definition 2.1.19 (NHIRSD Distribution)**.** *For positive integers n, w_1, and w_2, let $P \in \mathbb{F}_q[X]$ be an irreducible polynomial of degree n, let $S(n,3)$ be the set of the parity-check matrices $\mathbf{H}$ under systematic form of 3-ideal codes of type $[3n,n]$ (see Equation 1). The* NHIRSD (n,w_1,w_2) Distribution *chooses uniformly at random a matrix $\mathbf{H} \overset{\$}{\leftarrow} S(n,3)$ together with a vector $\mathbf{x} = (\mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_3) \overset{\$}{\leftarrow} \mathbb{F}_{q^m}^{3n}$ such that $\|(\mathbf{x}_1,\mathbf{x}_3)\| = w_1$, $\|\mathbf{x}_2\| = w_1 + w_2$,* $\mathrm{Supp}(\mathbf{x}_1,\mathbf{x}_3) \subset \mathrm{Supp}(\mathbf{x}_2)$*, and outputs $(\mathbf{H}, \mathbf{H}\mathbf{x}^\top)$.*

Definition 2.1.20 (Computational NHIRSD Problem). *For positive integers n, w_1, and w_2, let $P \in \mathbb{F}_q[X]$ be an irreducible polynomial of degree n, let $\mathbf{H}$ be a random parity check matrix under systematic form of an 3-ideal code $\mathcal{C}$ and $\mathbf{y} \overset{\$}{\leftarrow} \mathbb{F}_{q^m}^{2n}$. The* Computational non-homogeneous 3-ideal RSD Problem NHIRSD (n, w_1, w_2) *asks to find $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3) \overset{\$}{\leftarrow} \mathbb{F}_{q^m}^{3n}$ such that $\|(\mathbf{x}_1, \mathbf{x}_3)\| = w_1$, $\|\mathbf{x}_2\| = w_1 + w_2$, $\mathrm{Supp}(\mathbf{x}_1, \mathbf{x}_3) \subset \mathrm{Supp}(\mathbf{x}_2)$, and $\mathbf{y} = \mathbf{x}\mathbf{H}^\top$.*

Proposition 2.1.1. *For positive integers n, w_1, and w_2, let $P \in \mathbb{F}_q[X]$ be an irreducible polynomial of degree n. The Search 3-IRSD (n, w_1) problem reduces to the Search NHIRSD (n, w_1, w_2) problem.*

Proof. The proof is straightforward, if one has an algorithm $\mathcal{A}(\eta, \omega_1, \omega_2)$ which can output a solution to a NHIRSD instance of arbitrary parameters $(\eta, \omega_1, \omega_2)$, then given a 3-IRSD instance of parameters (n', w'_1), one solves it calling $\mathcal{A}$ with parameters $(n', w'_1, 0)$ and outputs its solution. □

Definition 2.1.21 (Decisional NHIRSD Problem). *For positive integers n, w_1, and w_2, let $P \in \mathbb{F}_q[X]$ be an irreducible polynomial of degree n, let $S(n, 3)$ be the set of the parity-check matrices $\mathbf{H}$ under systematic form of 3-ideal codes of type $[3n, n]$ (see Equation 1). The* Decisional non-homogeneous 3-ideal RSD Problem DNHIRSD (n, w_1, w_2) *asks to decide with non-negligible advantage whether $(\mathbf{H}, \mathbf{y}^\top)$ came from the NHIRSD (n, w_1, w_2) distribution or the uniform distribution over $S(n, 3) \times \mathbb{F}_{q^m}^{2n}$.*

2.1.5 Encryption and security

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

- Setup(1^λ), where λ is the security parameter, generates the global parameters param of the scheme;
- KeyGen(param) outputs a pair of keys, a (public) encryption key pk and a (private) decryption key sk;
- Encrypt$(\mathsf{pk}, \mathbf{m}, \theta)$ outputs a ciphertext $\mathbf{c}$, from the message $\mathbf{m}$, under the encryption key pk using randomness θ;
- 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 param $\leftarrow$ Setup(1^λ), every pair of keys (pk, sk) generated by KeyGen, every message $\mathbf{m}$, we should have

$$P[\mathsf{Decrypt}(\mathsf{sk}, \mathsf{Encrypt}(\mathsf{pk}, \mathbf{m}, \theta)) = \mathbf{m}] = 1 - \mathsf{negl}(\lambda),$$

where $\mathsf{negl}(\cdot)$ is a negligible function and where the probability is taken over varying randomness.

IND-CPA [13]: 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:

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{E}}(\lambda, t) = \max_{|\mathcal{A}| \leq t} \mathsf{Adv}^{\mathsf{ind}}_{\mathcal{E},\mathcal{A}}(\lambda), \tag{3}$$

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

```
$\mathbf{Exp}^{\mathsf{ind}-b}_{\mathcal{E},\mathcal{A}}(\lambda)$
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. $\mathbf{c}^* \leftarrow \mathsf{Encrypt}(\mathsf{pk}, \mathbf{m}_\mathbf{b}, \theta)$
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.

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{E},\mathcal{A}}(\lambda) = \left|\Pr[\mathbf{Exp}^{\mathsf{ind}-1}_{\mathcal{E},\mathcal{A}}(\lambda) = 1] - \Pr[\mathbf{Exp}^{\mathsf{ind}-0}_{\mathcal{E},\mathcal{A}}(\lambda) = 1]\right|. \tag{4}$$

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 [15] for our scheme to pass from IND-CPA to IND-CCA2.

In [15] 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 [15] 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. Let us denote by $\mathsf{Encrypt}(\mathsf{pk}, \mathbf{m}, \theta)$ an encryption function that relies on θ to generate random values. The idea of [15] 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 [15] for more details).

2.2 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 [15] 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. 2.4.

Notation: $\mathcal{S}_w^n(\mathbb{F}_{q^m})$ stands for the set of vectors of length n and rank weight w over $\mathbb{F}_{q^m}$, $\mathcal{S}_{1,w}^n(\mathbb{F}_{q^m})$ stands for the set of vectors of length n and rank weight w, *such that their support contains* 1, and $\mathcal{S}_{(w_1,w_2)}^{3n}(\mathbb{F}_{q^m})$ stands for the set of vectors of length $3n$ with *non-homogeneous* rank weight (w_1, w_2) over of $\mathbb{F}_{q^m}$, more precisely:

$$\begin{aligned}
\mathcal{S}_w^n(\mathbb{F}_{q^m}) &= \{\mathbf{x} \in \mathbb{F}_{q^m}^n : \|\mathbf{x}\| = w\} \\
\mathcal{S}_{1,w}^n(\mathbb{F}_{q^m}) &= \{\mathbf{x} \in \mathbb{F}_{q^m}^n : \|\mathbf{x}\| = w, 1 \in \mathrm{Supp}(\mathbf{x})\} \\
\mathcal{S}_{(w_1,w_2)}^{3n}(\mathbb{F}_{q^m}) &= \{\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3) \in \mathbb{F}_{q^m}^{3n} : \|(\mathbf{x}_1, \mathbf{x}_3)\| = w_1, \|\mathbf{x}_2\| = w_1 + w_2, \\
&\qquad \mathrm{Supp}(\mathbf{x}_1, \mathbf{x}_3) \subset \mathrm{Supp}(\mathbf{x}_2)\}
\end{aligned}$$

2.2.1 Public key encryption version (RQC.PKE)

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

- $\mathsf{Setup}(1^\lambda)$: generates and outputs the global parameters $\mathsf{param} = (n, k, \delta, w, w_1, w_2, P)$ where $P \in \mathbb{F}_q[X]$ is an irreducible polynomial of degree n.
- $\mathsf{KeyGen}(\mathsf{param})$: samples $\mathbf{h} \overset{\$}{\leftarrow} \mathbb{F}_{q^m}^n$, $\mathbf{g} \overset{\$}{\leftarrow} \mathcal{S}_n^n(\mathbb{F}_{q^m})$ and $(\mathbf{x}, \mathbf{y}) \overset{\$}{\leftarrow} \mathcal{S}_{1,w}^{2n}(\mathbb{F}_{q^m})$, computes the generator matrix $\mathbf{G} \in \mathbb{F}_{q^m}^{k \times n}$ of $\mathcal{G}_{\mathbf{g}}(n, k, m)$, sets $\mathsf{pk} = (\mathbf{g}, \mathbf{h}, \mathbf{s} = \mathbf{x} + \mathbf{h} \cdot \mathbf{y} \mod P)$ and $\mathsf{sk} = (\mathbf{x}, \mathbf{y})$, returns $(\mathsf{pk}, \mathsf{sk})$.
- $\mathsf{Encrypt}(\mathsf{pk}, \mathbf{m}, \theta)$: uses randomness θ to generate $(\mathbf{r}_1, \mathbf{e}, \mathbf{r}_2) \overset{\$}{\leftarrow} \mathcal{S}_{(w_1,w_2)}^{3n}(\mathbb{F}_{q^m})$, sets $\mathbf{u} = \mathbf{r}_1 + \mathbf{h} \cdot \mathbf{r}_2 \mod P$ and $\mathbf{v} = \mathbf{mG} + \mathbf{s} \cdot \mathbf{r}_2 + \mathbf{e} \mod P$, returns $\mathbf{c} = (\mathbf{u}, \mathbf{v})$.
- $\mathsf{Decrypt}(\mathsf{sk}, \mathbf{c})$: returns $\mathcal{G}_{\mathbf{g}}.\mathsf{Decode}(\mathbf{v} - \mathbf{u} \cdot \mathbf{y} \mod P)$.

Figure 2: Description of our proposal RQC.PKE.

Notice that the generator matrix $\mathbf{G}$ of the code $\mathcal{G}_{\mathbf{g}}$ 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 $\mathcal{G}_{\mathbf{g}}$ being used.

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

$$\mathsf{Decrypt}\left(\mathsf{sk}, \mathsf{Encrypt}\left(\mathsf{pk}, \mathbf{m}, \theta\right)\right) = \mathbf{m}. \tag{5}$$

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

$$\|\mathbf{s} \cdot \mathbf{r}_2 - \mathbf{u} \cdot \mathbf{y} + \mathbf{e}\| \leq \delta \tag{6}$$

$$\|\left(\mathbf{x} + \mathbf{h} \cdot \mathbf{y}\right) \cdot \mathbf{r}_2 - \left(\mathbf{r}_1 + \mathbf{h} \cdot \mathbf{r}_2\right) \cdot \mathbf{y} + \mathbf{e}\| \leq \delta \tag{7}$$

$$\|\mathbf{x} \cdot \mathbf{r}_2 - \mathbf{r}_1 \cdot \mathbf{y} + \mathbf{e}\| \leq \delta \tag{8}$$

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. 2.4.

2.2.2 KEM/DEM version (RQC.KEM)

Let $\mathcal{E}$ be an instance of the RQC.PKE cryptosystem as described above. Let $\mathcal{G}$, $\mathcal{H}$, and $\mathcal{K}$ be hash functions. The KEM-DEM version of the RQC cryptosystem is described in Figure 3.

According to [15], the RQC.KEM is IND-CCA2. More details regarding the tightness of the reduction are provided at the end of Sec. 2.4.

Security concerns and implementation details. Notice that while NIST only recommends SHA512 as a hash function, the transformation of [15] 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 the secret key of $\mathcal{E}$. We therefore suggest to use SHA3-512 for $\mathcal{G}$ and SHA512 for $\mathcal{H}$.

- $\mathsf{Setup}(1^\lambda)$: 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 $\mathbf{m} \overset{\$}{\leftarrow} \mathbb{F}_{q^m}^k$ (this will serve as a seed to derive the shared key). Derive the randomness $\theta \leftarrow \mathcal{G}(\mathbf{m})$. Generate the ciphertext $\mathbf{c} \leftarrow (\mathbf{u}, \mathbf{v}) = \mathcal{E}.\mathsf{Encrypt}(\mathsf{pk}, \mathbf{m}, \theta)$, 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.

2.2.3 A hybrid encryption scheme (RQC.HE)

NIST announced that they will be using generic transformations to convert any IND-CCA2 KEM into an IND-CCA2 PKE although 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.

2.3 Representation of objects

Field elements. Elements of $\mathbb{F}_{q^m}$ are represented as vectors of size m over $\mathbb{F}_q$. For RQC, q is chosen equal to 2 (see section 2.4) thus $e \in \mathbb{F}_{q^m}$ is represented as $(e_0, \ldots, e_{m-1}) \in \mathbb{F}_2^m$. In the reference implementation, elements are stored using $8 \times \lceil m/64 \rceil$ bytes in which the unused $64 \times \lceil m/64 \rceil - m$ bits are zero-padded. In the optimized implementation, elements are stored using $16 \times \lceil m/128 \rceil$ bytes in which the unused $128 \times \lceil m/128 \rceil - m$ are zero-padded. The first bit e_0 corresponds to the constant coefficient of the polynomial e.

Vectors. Elements of $\mathbb{F}_{q^m}^n$ are represented as n-dimensional arrays of $\mathbb{F}_{q^m}$ elements.

Seeds. The considered seed-expander has been provided by the NIST. It is initialized with a byte string of length 40 of which 32 are used as the `seed` and 8 are used as the `diversifier`. In addition, it is initialized with `max_length` equal to $2^{32} - 1$.

2.3.1 Parsing vectors from/to byte strings

Vectors of $\mathbb{F}_{q^m}^n$ are converted to byte strings using a compact representation in which the unused bits of each element are removed, thus leading to a $\lceil nm/8 \rceil$ long byte string. The compact representation is used for the public key pk, the secret key sk, the ciphertext $\mathbf{c}$ and for the inputs of the hash functions $\mathcal{G}, \mathcal{H}$ and $\mathcal{K}$.

2.3.2 Keys and ciphertext representation

The secret key $\mathsf{sk} = (\mathbf{x}, \mathbf{y})$ is represented as $\mathsf{sk} = (\mathbf{seed1})$ where $\mathbf{seed1}$ is used to generate $\mathbf{x}$ and $\mathbf{y}$. The public key $\mathsf{pk} = (\mathbf{g}, \mathbf{h}, \mathbf{s})$ is represented as $\mathsf{pk} = (\mathbf{seed2}, \mathbf{s})$ where $\mathbf{seed2}$ is used to generate $\mathbf{g}$ and $\mathbf{h}$. The generator matrix $\mathbf{G}$ is generated from $\mathbf{g}$ with respect to Definition 2.1.8. The ciphertext $\mathbf{c}$ is represented as $(\mathbf{u}, \mathbf{v}, \mathbf{d})$ where $\mathbf{d}$ is generated using SHA512. The secret key has size 40 bytes, the public key has size $40 + \lceil nm/8 \rceil$ bytes, and the ciphertext has size $2\lceil nm/8 \rceil + 64$ bytes.

2.3.3 Randomness and vector generation

Random bytes are generated using the NIST provided `randombytes` or `seedexpander` functions. The `randombytes` function is used to generate $\mathbf{seed1}$ and $\mathbf{seed2}$ as well as $\mathbf{m}$. The `seedexpander` function is used to generate θ (using $\mathbf{m}$ as seed) as well as $\mathbf{x}$, $\mathbf{y}$ (using $\mathbf{seed1}$ as seed), $\mathbf{g}$, $\mathbf{h}$ (using $\mathbf{seed2}$ as seed) and $\mathbf{r}_1$, $\mathbf{r}_2$, $\mathbf{e}$ (using θ as seed).

Random vectors are sampled uniformly from $\mathbb{F}_{q^m}^k$, $\mathbb{F}_{q^m}^n$, $\mathcal{S}_n^n(\mathbb{F}_{q^m})$, $\mathcal{S}_{1,w}^{2n}(\mathbb{F}_{q^m})$ or $\mathcal{S}_{(w_1,w_2)}^{3n}(\mathbb{F}_{q^m})$. Sampling from $\mathbb{F}_{q^m}^k$ and $\mathbb{F}_{q^m}^n$ is performed by filling the mathematical representation of the vector with random bits. Sampling from $\mathcal{S}_n^n(\mathbb{F}_{q^m})$ uses the same process and repeat it until a full rank vector is found. Sampling from $\mathcal{S}_{1,w}^{2n}(\mathbb{F}_{q^m})$ starts by generating a full rank support vector of size w that contains 1. Next, the sampled vector is generated by setting the coordinates of the support vector at random positions and setting the remaining coordinates as random linear combinations of the support vector coordinates. Sampling from $\mathcal{S}_{(w_1,w_2)}^{3n}(\mathbb{F}_{q^m})$ uses a similar process with two full rank support vectors of size w_1 and $w_1 + w_2$ respectively. The support vector of size $w_1 + w_2$ is generated first and its w_1 first coordinates are used to create the support vector of size w_1.

2.4 Parameters

2.4.1 Error distribution and decoding algorithm: no decryption failure

The decryption algorithm of RQC requires 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_1) and $\mathbf{e}$ has rank weight $w_1 + w_2$. 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_1 and is in any case bounded from above by ww_1. 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 $w_1 + w_2$. So that overall the error $\mathbf{e}'$ to decode for decryption has a rank weight upper bounded by $(w + 1)w_1 + w_2$.

Now since we choose the secret vector $(\mathbf{x}, \mathbf{y})$ such that its support is a random subspace of $\mathbb{F}_{q^m}$ of dimension w containing 1, the weight of $\mathbf{e}'$ is upper bounded by $ww_1 + w_2$. Indeed, in this case, a part of 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 $\mathbb{F}_{q^m}$, which can decode $\frac{n-k}{2}$ rank errors and choose our parameters such that $ww_1 + w_2 \leq \frac{n-k}{2}$, so that *there is no decryption failure.*

2.4.2 Parameters and tightness of the reduction

In what follows, unless otherwise mentioned, DIRSD will refer to the 2-DIRSD problem. We also recall that the DNHIRSD problem always deals with ideal $[3n, n]$-code with non-homogeneous supports for the error which has rank weight (w_1, w_2).

The practical security of the public key of our scheme relies on the DIRSD problem for a small weight vector of weight $w = \|\mathbf{x}\| = \|\mathbf{y}\|$ with $w = \mathcal{O}(\sqrt{n})$.

The IND-CPA security of the scheme could be reduced to the DIRSD and the DNHIRSD problems for decoding, respectively, a random ideal $[2n, n]$-code for a vector $(\mathbf{r}_1, \mathbf{r}_2)$ with small weight w_1 and a random ideal $[3n, n]$-code for a vector $(\mathbf{r}_1, \mathbf{e}, \mathbf{r}_2)$ with small *non-homogeneous* weight (w_1, w_2).

There are two kinds of attacks against those problems: the combinatorial ones and the algebraic ones.

Combinatorial attacks. The current best known combinatorial attacks are given in [11, 3] and their complexity is detailed in Section 6.

For a given block length n and a rank weight w_1, the 3-DIRSD (n, w_1) problem is easier to solve than the 2-DIRSD (n, w_1) problem and the 3-DIRSD (n, w_1) problem reduces to the DNHIRSD (n, w_1, w_2) problem (Proposition 2.1.1).

In practice, for our parameters, the combinatorial attacks are less efficient than the algebraic attacks.

Algebraic attacks. The current best known algebraic attacks are described in [5, 6] and their complexity are detailed in Section 6.

For algebraic attacks, the additional support of $\mathbf{e}$ of rank weight w_2 in the DNHIRSD(n, w_1, w_2) problem does have an impact on its complexity, this is discussed in Section 6 using an adaptation of [6] to non-homogeneous error. In other words, the bigger the rank weight w_2, the harder to solve DNHIRSD (n, w_1, w_2), so this additional support gives us a way to control the complexity of the algebraic attacks against it.

Thus, we chose the value of w_1 and w_2 such that the parameters fits the required security complexities; essentially, for our parameters, the complexity of the algebraic attacks against DNHIRSD (n, w_1, w_2) is greater than for 2-DIRSD (n, w_1), except for the parameters of RQC-III-256 for which the complexity of DNHIRSD (n, w_1, w_2) is a little below 2-DIRSD (n, w_1), but still greater than 256.

Quantum attacks. At the current state of research, there is no real quantum speed-up for the aforementioned algebraic attacks on which our parameters are largely based.

The best quantum attacks on the rank metric problems follow [10], in that case there is a square root gain on the probabilistic part of the attack (details are given in [10]).

For instance, from a quantum point of view, RQC-I-192 has a security of 128 quantum-bits.

2.4.3 Choice of parameters

Overall, the parameters proposed in Tab. 1 correspond to tight reduction for generic instances of the DIRSD problem in the rank metric.

All of our submissions use ideal code over $\mathbb{F}_{2^m}$ in order to reduce the size of the key and to allow to compute the syndrome of an error as sums and products of polynomials in $\mathbb{F}_{2^m}[X]/\langle P\rangle$, with $P \in \mathbb{F}_2[X]$ of degree n. In order to avoid folding attacks (see [14]), P is chosen irreducible. Moreover, to decrease the computational costs, we want P to be sparse. We have obtained these polynomials with the Magma software. More details are available at `http://magma.maths.usyd.edu.au/magma/handbook/text/193#1685`. Tab. 2 presents the aforementioned polynomials as well as the polynomials used to define $\mathbb{F}_{2^m}$.

The decoding Gabidulin code has length n, dimension k over $\mathbb{F}_{q^m}$ and corrects errors of weight up to $(n-k)/2 \geqslant ww_1 + w_2$. The resulting public key, secret key, ciphertext and shared secret sizes are given 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_1	w_2	Security
RQC-I	2	127	113	3	7	7	6	128
RQC-II	2	151	149	5	8	8	8	192
RQC-III	2	181	179	3	9	9	7	256

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

Instance	P	Π
RQC-I	$X^{113} + X^9 + 1$	$X^{127} + X + 1$
RQC-II	$X^{149} + X^{10} + X^9 + X^7 + 1$	$X^{151} + X^3 + 1$
RQC-III	$X^{179} + X^4 + X^2 + X + 1$	$X^{181} + X^7 + X^6 + X + 1$

Table 2: Polynomials considered for RQC. P is the polynomial used to define $\mathbb{F}_{q^m}^n$ as $\mathbb{F}_{q^m}[X]/\langle P\rangle$ and Π is the polynomial used to define $\mathbb{F}_{q^m}$ as $\mathbb{F}_q[X]/\langle \Pi\rangle$.

Instance	pk size	sk size	ct size	ss size	Security
RQC-I	1834	40	3652	64	128
RQC-II	2853	40	5690	64	192
RQC-III	4090	40	8164	64	256

Table 3: Sizes in bytes for RQC (see section 2.3). The security is expressed in bits.

3 Performance Analysis

This section provides performance measures of our implementations of RQC.KEM.

Benchmark platform. The benchmarks have been performed on a machine that has 16GB of memory and an Intel® Core™ i7-7820X CPU @ 3.6GHz for which the Hyper-Threading, Turbo Boost and SpeedStep features were disabled. The scheme have been compiled with `gcc` (version 9.2.0) and use the `openssl` (version 1.1.1d) library as a provider for SHA2. For each parameter set, the results have been obtained by computing the mean from 1000 random instances. In order to minimize biases from background tasks running on the benchmark platform, each instances have been repeated 100 times and averaged.

Constant time. The provided implementations have been implemented in a constant time way whenever relevant and as such their running time should not leak any information with respect to sensible data. For instance, such sensible data include secret keys as well as the weight of the error to be decoded by the Gabidulin code (see section 6.4).

3.1 Reference Implementation

The performances of our reference implementation on the aforementioned benchmark platform are described in Tab. 4. The following optimization flags have been used during compilation: `-O3 -flto`.

Instance	KeyGen	Encaps	Decaps
RQC-128	0.87	1.62	10.42
RQC-192	1.89	3.63	22.26
RQC-256	2.86	5.27	36.39

Table 4: Millions of CPU cycles of RQC reference implementation.

3.2 Optimized Implementation

An optimized implementation leveraging AVX and CLMUL instructions have been provided. Its performances on the aforementioned benchmark platform are described in Tab. 5. The following optimization flags have been used during compilation: `-O3 -flto -mavx2 -mpclmul -msse4.2 -maes`.

Instance	KeyGen	Encaps	Decaps
RQC-128	0.37	0.53	2.58
RQC-192	0.76	1.16	5.65
RQC-256	1.15	1.71	9.35

Table 5: Millions of CPU cycles of RQC reference implementation.

4 Known Answer Test Values

Known Answer Test (KAT) values have been generated using the script provided by the NIST and can be retrieved in the `KAT/Reference_Implementation/` and `KAT/Optimized_Implementation` folders. In addition, examples with *intermediate values* have also been provided in these folders.

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.

5 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 [15], and the tightness of the reduction provided by this transformation has been discussed at the end of Sec. 2.1.5.

Theorem 5.1. *The scheme presented above is* IND-CPA *under the DIRSD and DNHIRSD 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 one can build a simulator running in approximately the same time that breaks the DIRSD assumption for 2-ideal codes or the DNHIRSD assumption for 3-ideal codes.

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 $\mathsf{pk} = (\mathbf{g}, \mathbf{h}, \mathbf{s} = \mathbf{x} + \mathbf{h} \cdot \mathbf{y})$ and $\mathsf{sk} = (\mathbf{x}, \mathbf{y})$
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. $\mathbf{b}' \leftarrow \mathcal{A}(\mathtt{GUESS} : \mathbf{c}^*)$
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 $\mathsf{pk} = (\mathbf{g}, \mathbf{h}, \mathbf{s} = \mathbf{x} + \mathbf{h} \cdot \mathbf{y})$ and $\mathsf{sk} = (\mathbf{x}, \mathbf{y})$
2b. $\mathbf{s} \overset{\$}{\leftarrow} \mathbb{F}^n_{q^m}$
2c. $(\mathsf{pk}, \mathsf{sk}) \leftarrow ((\mathbf{g}, \mathbf{h}, \mathbf{s}), \mathbf{0})$
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. $\mathbf{b}' \leftarrow \mathcal{A}(\mathtt{GUESS} : \mathbf{c}^*)$
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 DIRSD (n, w) problem for parameters (n, w) resulting from $\mathsf{Setup}(1^\lambda)$, with the same advantage ϵ, when given as input a challenge $(\mathbf{H}, \mathbf{y}^\top) \in \mathbb{F}^{n\times 2n}_{q^m} \times \mathbb{F}^n_{q^m}$.

```
$\mathcal{D}'_{\mathcal{E},\mathcal{D}_\lambda}((\mathbf{H}, \mathbf{y}^\top))$
1. Set $\mathsf{param} \leftarrow \mathsf{Setup}(1^\lambda)$
2a. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{param})$ with $\mathsf{pk} = (\mathbf{g}', \mathbf{h}', \mathbf{s}' = \mathbf{x}' + \mathbf{h}' \cdot \mathbf{y}')$ and $\mathsf{sk} = (\mathbf{x}', \mathbf{y}')$
2b. $(\mathsf{pk}, \mathsf{sk}) \leftarrow ((\mathbf{g}', \mathbf{h}, \mathbf{y}), \mathbf{0})$
3. $\mathbf{b}' \leftarrow \mathcal{D}_\lambda(\mathsf{pk})$
4. If $\mathbf{b}' == 1$ output IRSD
5. If $\mathbf{b}' == 2$ output UNIFORM
```

Note that if we define pk as $(\mathbf{g}', \mathbf{h}, \mathbf{y})$ with $\mathbf{g}'$ generated by $\mathsf{KeyGen}(\mathsf{param})$ and $(\mathbf{H}, \mathbf{y}^\top)$ from a IRSD (n, w) 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, then pk follows exactly the same distribution as in Game $\mathbf{G}_2$. Thus we have:

$$Pr\left[\mathcal{D}'_{\mathcal{E},\mathcal{D}_\lambda}((\mathbf{H}, \mathbf{y}^\top)) = \mathtt{IRSD} | (\mathbf{H}, \mathbf{y}^\top) \leftarrow 2\text{-}\mathsf{IRSD}(n, w)\right] = Pr\left[\mathcal{D}_\lambda(\mathsf{pk}) = 1 | \mathsf{pk} \text{ from } \mathbf{Game}^1_{\mathcal{E},\mathcal{A}}(\lambda)\right]$$

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

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 $\mathbf{Game}^2_{\mathcal{E},\mathcal{A}}(\lambda)$. 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 $\mathsf{pk} = (\mathbf{g}, \mathbf{h}, \mathbf{s} = \mathbf{x} + \mathbf{h} \cdot \mathbf{y})$ and $\mathsf{sk} = (\mathbf{x}, \mathbf{y})$
2b. $\mathbf{s} \overset{\$}{\leftarrow} \mathbb{F}^n_{q^m}$
2c. $(\mathsf{pk}, \mathsf{sk}) \leftarrow ((\mathbf{g}, \mathbf{h}, \mathbf{s}), \mathbf{0})$
3. $(\mathbf{m}_0, \mathbf{m}_1) \leftarrow \mathcal{A}(\mathtt{FIND} : \mathsf{pk})$
4a. Use randomness $\theta$ to generate $\mathbf{e} \overset{\$}{\leftarrow} \mathbb{F}^n_{q^m}$, $\mathbf{r} = (\mathbf{r}_1, \mathbf{r}_2) \overset{\$}{\leftarrow} \mathbb{F}^{2n}_{q^m}$ uniformly at random
4b. $\mathbf{u} \leftarrow \mathbf{r}_1 + \mathbf{h} \cdot \mathbf{r_2}$ 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

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

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

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

follows the NHIRSD distribution (for a $[3n, n]$ ideal code), 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 $b' \in \{2, 3\}$ of the game we are in.

Again, we can build a distinguisher $\mathcal{D}'_{\mathcal{E},\mathcal{D}_\lambda}$ that will break the $\mathsf{DNHIRSD}$ (n, w_1, w_2) assumption for parameters (n, w_1, w_2) from $\mathsf{Setup}(1^\lambda)$ with the same advantage as the game distinguisher, when given an input $(\mathbf{H}, \mathbf{y}^\top) \in \mathbb{F}_{q^m}^{2n\times 3n} \times \mathbb{F}_{q^m}^{2n}$. In the $\mathsf{DNHIRSD}$ (n, w_1, w_2) problem, matrix $\mathbf{H}$ is assumed to be of the form

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

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 $\mathbf{y} = (\mathbf{y_1}, \mathbf{y_2})$.

```
$\mathcal{D}'_{\mathcal{E},\mathcal{D}_\lambda}((\mathbf{H}_{\mathbf{a},\mathbf{b}}, (\mathbf{y_1}, \mathbf{y_2})^\top))$
1. $\mathsf{param} \leftarrow \mathsf{Setup}(1^\lambda)$
2a. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{param})$ with $\mathsf{pk} = (\mathbf{g}, \mathbf{h}, \mathbf{s} = \mathbf{x} + \mathbf{h} \cdot \mathbf{y})$ and $\mathsf{sk} = (\mathbf{x}, \mathbf{y})$
2b. $(\mathsf{pk}, \mathsf{sk}) \leftarrow ((\mathbf{g}, \mathbf{a}, \mathbf{b}), \mathbf{0})$
3. $(\mathbf{m}_0, \mathbf{m}_1) \leftarrow \mathcal{D}_\lambda(\texttt{FIND} : \mathsf{pk})$
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. $\mathbf{b}' \leftarrow \mathcal{D}_\lambda(\texttt{GUESS} : \mathbf{c}^*)$
4. If $\mathbf{b}' == 2$ output IRSD
5. If $\mathbf{b}' == 3$ output UNIFORM
```

The distribution of pk is unchanged with respect to the games as $\mathbf{g}$ is generated by $\mathsf{KeyGen}(\mathsf{param})$ and $\mathbf{a}$ and $\mathbf{b}$ are both uniformly chosen. If $(\mathbf{H}_{\mathbf{a},\mathbf{b}}, (\mathbf{y_1}, \mathbf{y_2})^\top)$ follows the $\mathsf{DNHIRSD}$ (n, w_1, w_2) distribution, then

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

with $\|(\mathbf{x}_1, \mathbf{x}_3)\| = w_1$, $\|\mathbf{x}_2\| = w_1 + w_2$, and $\mathrm{Supp}((\mathbf{x}_1, \mathbf{x}_3)) \subset \mathrm{Supp}(\mathbf{x}_2)$. 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 at random and set $\mathbf{u} = \mathbf{r}'_1 + \mathbf{h} \cdot \mathbf{r}'_2$ 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 $\mathsf{pk} = (\mathbf{g}, \mathbf{h}, \mathbf{s} = \mathbf{x} + \mathbf{h} \cdot \mathbf{y})$ and $\mathsf{sk} = (\mathbf{x}, \mathbf{y})$
2b. $\mathbf{s} \overset{\$}{\leftarrow} \mathbb{F}^n_{q^m}$
2c. $(\mathsf{pk}, \mathsf{sk}) \leftarrow ((\mathbf{g}, \mathbf{h}, \mathbf{s}), \mathbf{0})$
3. $(\mathbf{m}_0, \mathbf{m}_1) \leftarrow \mathcal{A}(\mathtt{FIND} : \mathsf{pk})$
4a. Use randomness $\theta$ to generate $\mathbf{e}' \overset{\$}{\leftarrow} \mathbb{F}^n_{q^m}$ and $\mathbf{r}' = (\mathbf{r}'_1, \mathbf{r}'_2) \overset{\$}{\leftarrow} \mathbb{F}^{2n}_{q^m}$
4b. $\mathbf{u} \leftarrow \mathbf{r}'_1 + \mathbf{h} \cdot \mathbf{r}'_2$ 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. $\mathbf{b}' \leftarrow \mathcal{A}(\mathtt{GUESS} : \mathbf{c}^*)$
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 $\mathbf{e}' = \mathbf{e} + \mathbf{m}_0\mathbf{G} - \mathbf{m}_1\mathbf{G}$. 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 DNHIRSD 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 DIRSD 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:

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{E},\mathcal{A}}(\lambda) \leq 2\left(\mathsf{Adv}^{\mathsf{DIRSD}}(\lambda) + \mathsf{Adv}^{\mathsf{DNHIRSD}}(\lambda)\right). \tag{9}$$

□

6 Known Attacks

In this section, we present the best known attacks against the DIRSD (Definition 2.1.18) and DNHIRSD (Definition 2.1.21) problems on which RQC is based.

Although these problems are decisional problems, the best attacks in the current state of research rely on solving their computational version. There exist two types of attacks on these problems:

- combinatorial attacks where the goal is to find the support of the error or of the codeword ;
- algebraic attacks where the opponent tries to solve an algebraic system, for instance using Gröbner basis computation.

First, we will deal with combinatorial attacks, and then we will discuss algebraic attacks. In addition to this, we will also provide some details regarding timing attacks against RQC.

These attacks are generic attacks against the RSD and NHRSD problems since there is no known improvement which exploits the ideal structure of the codes.

6.1 Combinatorial attacks

The best combinatorial attack to solve the RSD problem (Definition 2.1.14) for an $[sn, n]$-code over $\mathbb{F}_{q^m}$ with an error of rank weight r has a complexity of:

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

operations in $\mathbb{F}_q$, where ω is the exponent of the complexity to find the solution of a linear system.

This attack is an improvement of a previous attack described in [11], 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 the Hamming distance. For the rank metric, the attacker tries and guesses a subspace which contains the support of the error and then solves a linear system obtained from the parity-check equations to check if the choice was correct.

Remark 6.1. *Since the linear system is not random, it is reasonable to take $\omega = 2$ for the choice of the parameters of RQC, even if the attack described in [3] takes $\omega = 3$. Let us remark that the choice of our parameter is flexible. We could take $\omega = 0$ and increase the parameters, this would correspond to only keeping the exponential complexity of the attack.*

6.2 Algebraic attacks

This section, except the very last part (*underdetermined case* for the NHRSD problem), is taken from [6] with some minors modifications.

6.2.1 Algebraic attack against RSD

The second way to solve an RSD instance is to write it as a system of equations, called a *modeling*, then, if one solves this system, that is to say finds one solution, it is a solution to the RSD instance.

Ourivski and Johansson pioneered the algebraic attack against RSD by giving a modeling in [20], then Levy and Perret first proposed to solve it using Gröbner basis computations in [16]. In [5], a new modeling was proposed, it is based on maximal minors taken from a slightly different version of Ourivksi and Johansson's modeling.

From now on, in this section, we deal with RSD instances based on $[n, k]$-code over $\mathbb{F}_{2^m}$ with target rank weight r.

The modeling uses the fact that the vector $\mathbf{e}$ of small weight r can be written as a product $\mathbf{SC}$ where $\mathbf{S}$ is a matrix containing a basis of the support of $\mathbf{e}$ and $\mathbf{C}$ is a matrix containing the coordinates of each component of $\mathbf{e}$ in this basis. Those matrices are called the *support* and the *coordinate* matrices.

Roughly, the system will consist in $m\binom{n-k-1}{r}$ equations in $\binom{n}{r}$ variables which are maximal minors, in [5], this system was then solved by computing its Gröbner basis; in [6], a different modeling enables one to solve it directly by linearization. More precisely, the condition

$$m\binom{n-k-1}{r} \geq \binom{n}{r} \tag{10}$$

is inherent to this system; when it is fulfilled, it corresponds to *the overdetermined case*, if it is not, it corresponds to *the underdetermined case*.

Overdetermined case. If the condition (10) is fulfilled, solving the RSD problem is equivalent to solving a linear system with $m\binom{n-k-1}{r}$ equations in $\binom{n}{r}$ variables, which can be done with a cost in

$$\mathcal{O}\left(m\binom{n-k-1}{r}\binom{n}{r}^{\omega-1}\right). \tag{11}$$

On the one hand, if the condition (10) is *widely* fulfilled, there is an optimization in [6] to reduce the complexity of (11) using a punctured version of the code. On the other hand, if (10) is not fulfilled, one can reduce to it by guessing few variables at an exponential cost, it is an hybrid attack, also in [6].

Underdetermined case. For cryptographic purpose, the parameters are chosen so that they belong to an area where the condition (10) is obviously not fulfilled and where the exponential cost of the hybrid attack would make it impractical, this particular area is called the *underdetermined case*. The best known complexity in this case is also described in [6]: it uses a variation of the aforementioned system together with a new setting coming from the reduction between RSD and the MinRank problem. This new system, despite being bigger, is sometimes sparser, so its resolution could take advantage of Wiedemann algorithm to solve sparse linear systems.

More precisely, the complexity in the underdetermined case is

$$\mathcal{O}\left((B_b + C_b)A_b^{\omega-1}\right) \tag{12}$$

where

$$\begin{aligned}
A_b &:= \sum_{j=1}^{b} \binom{n}{r}\binom{mk+1}{j} \\
B_b &:= \sum_{j=1}^{b} \left(m\binom{n-k-1}{r}\binom{mk+1}{j} \right) \\
C_b &:= \sum_{j=1}^{b}\sum_{i=1}^{j} \left((-1)^{i+1}\binom{n}{r+i}\binom{m+i-1}{i}\binom{mk+1}{j-i} \right).
\end{aligned}$$

and where b is the smallest positive integer so that the condition $A_b - 1 \leq B_c + C_b$ is fulfilled.

When $b \geq 2$, one uses Wiedemann algorithm, resulting in a new complexity of

$$\mathcal{O}\left(\frac{B_b\binom{k+r+1}{r} + C_b(mk+1)(r+1)}{B_b + C_b} \left(\sum_{j=1}^{b} \binom{n}{r}\binom{mk+1}{j} \right)^2 \right). \quad (13)$$

6.2.2 Algebraic attack against NHRSD

As a reminder, for a code of length n over $\mathbb{F}_{2^m}$ with an error vector of rank weight r, one writes the error as a product $\mathbf{SC}$ where $\mathbf{S}$ is a vector of length r with entries in $\mathbb{F}_{2^m}$ consisting in a basis of the support of the error and $\mathbf{C}$ is a $r \times n$ matrix with entries in $\mathbb{F}_2$ consisting in the coordinates of each component of the error in this basis.

From now on, in this section, we deal with $[3n, n]$-code over $\mathbb{F}_{2^m}$, so n is the size of a block and the dimension of the code, and no longer its length.

To visualize the particular structure of the non-homogeneous error, one wants to write it using block matrices. Thus, the error vector $(\mathbf{r}_1, \mathbf{e}, \mathbf{r_2})$ is written as a product of the two following matrices

$$\widetilde{\mathbf{S}} = \left[\, \mathbf{S}_1 \,\middle|\, \mathbf{S}_2 \,\right] \in \mathbb{F}_{2^m}^{w_1+w_2} \quad \text{and} \quad \widetilde{\mathbf{C}} = \left[\begin{array}{c|c|c} \mathbf{C}_1 & \mathbf{C}_2 & \mathbf{C}_3 \\ \hline \mathbf{0} & \mathbf{C}_2' & \mathbf{0} \end{array} \right] \in \mathbb{F}_2^{(w_1+w_2)\times(3n)}.$$

The matrix $\mathbf{S}_1$ is a basis of $\mathrm{Supp}((\mathbf{r}_1, \mathbf{r}_2))$ and $\widetilde{\mathbf{S}}$ is a basis of $\mathrm{Supp}(\mathbf{e})$, recall that $\mathrm{Supp}((\mathbf{r}_1, \mathbf{r}_2)) \subset \mathrm{Supp}(\mathbf{e})$.

The aforementioned attack relied on the condition (10) in which the right part of the inequality counts the number of distinct maximal minors in $\widetilde{\mathbf{C}}$. With this approach, the condition does not take advantage of the structure of $\widetilde{\mathbf{C}}$. In fact, as it contains two zero blocks, its number of maximal minors equal to zero is

$$M := \sum_{i=0}^{w_2-1} \binom{2n}{w_1+w_2-i}\binom{n}{i}.$$

So one gets a new condition:

$$m\binom{2n-1}{w_1+w_2} \geq \binom{3n}{w_1+w_2} - M - 1. \tag{14}$$

As for the attack against RSD, this condition yields to two different cases: the overdetermined and underdetermined cases.

Overdetermined case. This new condition (14) yields to a new complexity in the case where it is fulfilled and to a new hybrid approach if it is not.

When the condition (14) it is fulfilled, the new complexity is

$$\mathcal{O}\left(m\binom{2n-1}{w_1+w_2}\left(\binom{3n}{w_1+w_2} - M\right)^{\omega-1}\right).$$

If it is not, one wants to guess a columns of $\widetilde{\mathbf{C}}$ to perform an hybrid approach as mentioned above. One notices that once again the structure of $\widetilde{\mathbf{C}}$ can be used by the attacker, in fact the cost of the exponential part of the hybrid attack is reduced if one guesses columns for which the lower block is a zero block. Doing so, the exponential term will drop from $q^{a(w_1+w_2)}$ down to q^{aw_2}. To sum everything up, the cost of the new attack (both in the overdetermined case, i.e. when $a = 0$, and the hybrid case) is

$$\mathcal{O}\left(q^{aw_1}m\binom{2n-1}{w_1+w_2}\left(\binom{3n-a}{w_1+w_2} - \underbrace{\sum_{i=0}^{w_2-1}\binom{2n-a}{w_1+w_2-i}\binom{n}{i}}_{:=\ M_a}\right)^{\omega-1}\right)$$

where a is the smallest integer such that the following condition is fulfilled

$$m\binom{2n-1}{w_1+w_2} \geq \binom{3n-a}{w_1+w_2} - M_a - 1. \tag{15}$$

Underdetermined case. If the condition (14) is not fulfilled and if the aforementioned hybrid approach is impractical, one can adapt the attack in the *underdetermined case* described in Section 6.2.1 to the case of non-homogeneous error.

The core of the attack still relies on writing the non-homogeneous error as a product of block matrices to be able to count the number of maximal minors of $\widetilde{\mathbf{C}}$ which are equal to zero, recall that this number is

$$M := \sum_{i=0}^{w_2-1}\binom{2n}{w_1+w_2-i}\binom{n}{i}.$$

The fact that M maximal minors will be equal to zero will not affect the number of equations which remains $B_b + C_b$ where

$$B_b := \sum_{j=1}^{b} \left(m \binom{2n-1}{w_1+w_2} \binom{mn+1}{j} \right)$$

$$C_b := \sum_{j=1}^{b} \sum_{i=1}^{j} \left((-1)^{i+1} \binom{3n}{w_1+w_2+i} \binom{m+i-1}{i} \binom{mn+1}{j-i} \right),$$

but it will change the number of variables which drops down to

$$A_b := \sum_{j=1}^{b} \left(\binom{3n}{w_1+w_2} - M \right) \binom{mn+1}{j}.$$

Thus, like in Section 6.2.1, one can solve this linear system with a complexity of

$$\mathcal{O}\left(\min \left((B_b + C_b) A_b^{\omega-1}, \frac{B_b \binom{n+w_1+w_2+1}{w_1+w_2} + C_b(mn+1)(w_1+w_2+1)}{B_b + C_b} A_b^2 \right) \right)$$

where b is the smallest positive integer such that $A_b - 1 \leq B_b + C_b$. The right part in the minimum corresponds to the use of Wiedemann algorithm, which is more interesting when the system is sparse enough.

6.3 Quantum speed-up

At the current state of research, there is no important quantum speed-up for the aforementioned algebraic attacks.

For combinatorial attacks, the quantum speed-up is easy to analyze. According to [10], a slight generalization of Grover's quantum search algorithm allows to divide by a factor 2 the exponential complexity of the attacks. Thus, the complexity of the quantum combinatorial attack against the RSD problem is

$$\mathcal{O}\left(((s-1)nm)^{\omega} q^{\frac{1}{2}(r\lceil \frac{m(n+1)}{sn} \rceil - m)} \right).$$

6.4 Timing attacks

The resistance of RQC to timing attacks have been studied in [7]. This paper describes timing attacks that rely on a correlation between the weight of the error to be decoded and the running time of Gabidulin code's decoding algorithm. These attacks are of theoretical interest, nevertheless they are quite impractical in real situations as they require a huge number of requests to a timing oracle. The provided reference and optimized implementations are implemented in a way that should not leak the weight of the error to be decoded thus preventing the aforementioned timing attacks.

7 Advantages and Limitations

7.1 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;
- it features more attractive parameters than most of the Hamming based proposals.

The null decryption failure rate allows to achieve a tight reduction for the IND-CCA2 security of the KEM-DEM version through the recent transformation of [15].

7.2 Limitations

Rank metric has very nice features, but the use of rank metric for cryptographic purposes is not very old (1991). It may seem to be a limitation, but still in recent years there have been a lot of activities on understanding the inherent computational difficulty of the related problems. The combinatorial attacks are very well understood, and the recent results of [5, 6] permit to have a clear view on the complexity of algebraic attacks.

References

[1] Carlos Aguilar-Melchor, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, and Gilles Zémor. Efficient encryption from random quasi-cyclic codes. *IEEE Transactions on Information Theory*, 64(5):3927–3943, 2018. *7*

[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. *15*

[3] Nicolas Aragon, Philippe Gaborit, Adrien Hauteville, and Jean-Pierre Tillich. A new algorithm for solving the rank syndrome decoding problem. Vail, USA, 2018. IEEE. *22*, *30*

[4] Daniel Augot, Pierre Loidreau, and Gwezheneg Robert. Generalized Gabidulin codes over fields of any characteristic. *Des. Codes Cryptogr.*, 86(8):1807–1848, Aug 2018. *12*, *13*, *14*

[5] Magali Bardet, Pierre Briaud, Maxime Bros, Philippe Gaborit, Vincent Neiger, Olivier Ruatta, and Jean-Pierre Tillich. An algebraic attack on rank metric code-based cryptosystems. In *Advances in Cryptology - EUROCRYPT 2020 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020. Proceedings*, 2020. To appear, preprint available on https://arxiv.org/pdf/1910.00810.pdf. *3*, *22*, *31*, *35*

[6] Magali Bardet, Maxime Bros, Daniel Cabarcas, Philippe Gaborit, Ray Perlner, Daniel Smith-Tone, Jean-Pierre Tillich, and Javier Verbel. Algebraic attacks for solving the rank decoding and minrank problems without Gröbner basis, 2020. Preprint available on https://arxiv.org/pdf/2002.08322.pdf. *3*, *22*, *30*, *31*, *35*

[7] Slim Bettaieb, Loïc Bidoux, Philippe Gaborit, and Etienne Marcatel. Preventing timing attacks against RQC using constant time decoding of Gabidulin codes. In *10th International Conference on Post-Quantum Cryptography, PQCrypto 2019, Chongqing China, May 8-10*, 2019. https://pqc-rqc.org/doc/pqcrypto19-rqc.pdf. *34*

[8] Ernest Mukhamedovich Gabidulin. Theory of codes with maximum rank distance. *Problemy Peredachi Informatsii*, 21(1):3–16, 1985. *11*, *12*

[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. *10*

[10] 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. *22*, *34*

[11] Philippe Gaborit, Olivier Ruatta, and Julien Schrek. On the complexity of the rank syndrome decoding problem. *IEEE Trans. Inform. Theory*, 62(2):1006–1019, 2016. https://arxiv.org/pdf/1301.1026.pdf. *22*, *30*

[12] Philippe Gaborit and Gilles Zémor. On the hardness of the decoding and the minimum distance problems for rank codes. *IEEE Trans. Inform. Theory*, 62(12):7245–7252, 2016. https://arxiv.org/pdf/1404.3482.pdf. *14*

[13] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984. *17*

[14] 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*

[15] Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. A modular analysis of the fujisaki-okamoto transformation. In *Theory of Cryptography Conference*, pages 341–371. Springer, 2017. http://eprint.iacr.org/2017/604. *7*, *17*, *18*, *19*, *25*, *35*

[16] Françoise Levy–dit–Vehel and L. Perret. Algebraic decoding of rank metric codes. *Proceedings of YACC*, 2006. *31*

[17] Pierre Loidreau. Properties of codes in rank metric. *arXiv preprint cs/0610057*, 2006. https://arxiv.org/pdf/cs/0610057.pdf. *11*

[18] Pierre Loidreau. A Welch–Berlekamp like algorithm for decoding Gabidulin codes. In *Coding and cryptography*, pages 36–45. Springer, 2006. *12*, *13*, *14*

[19] Oystein Ore. On a special class of polynomials. *Trans. Amer. Math. Soc.*, 35(3):559–584, 1933. *11*, *12*

[20] Alexei V. Ourivski and Thomas Johansson. New technique for decoding codes in the rank metric and its cryptography applications. *Problems of Information Transmission*, 38(3):237–246, 2002. *31*

[21] 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. *11*

[22] Danilo Silva, Frank R. Kschischang, and Ralf Kotter. Communication over finite-field matrix channels. *IEEE Trans. Inform. Theory*, 56(3):1296–1305, 2010. *11*

A Signed statements by the submitters

NIST requires statements about the intellectual property of the present submission. While NIST clearly mentioned they require the original paper version of these statements, the authors estimated useful to include a digital copy of these statements in this document. The paper version of these statements will be provided directly to Dustin Moody (or any other NIST member) at the first PQC Standardization Conference.

The remainder of this submission consists of statements. Below is a list of the statements included.

Statement by each submitter. Each of the authors has such a statement included.

Statement by patent owners. Carlos Aguilar Melchor and Philippe Gaborit have a patent owner statement. This patent also involves the CNRS (french national center for scientific research), represented by Éric Buffenoir. Therefore a patent statement for the CNRS is also included.

Statement by reference/optimized implementations' owners. Each of the authors has such a statement included.