LRPC codes with multiple syndromes: near ideal-size KEMs without ideals

Carlos Aguilar-Melchor, Nicolas Aragon, Victor Dyseryn, Philippe Gaborit, Gilles Zémor

Monday, April 4, 2022
Plan

1. Background on rank metric and LRPC codes
2. Presentation of LRPC-MS
3. Analysis of the decoding failure rate
4. Bonus: advances in LRPC implementations
5. Conclusion and perspectives
Summary

1 Background on rank metric and LRPC codes
2 Presentation of LRPC-MS
3 Analysis of the decoding failure rate
4 Bonus : advances in LRPC implementations
5 Conclusion and perspectives
In rank metric, we consider $\mathbb{F}_{q^m}$-linear codes ($\mathbb{F}_{q^m}$ is a field extension of $\mathbb{F}_q$ of degree $m$).

**Definition (Rank weight)**

An element $\mathbf{x} = (x_1, ..., x_n) \in (\mathbb{F}_{q^m})^n$ can be unfold against an $\mathbb{F}_q$-basis of $\mathbb{F}_{q^m}$ in a matrix

$$
\mathcal{M}(\mathbf{x}) = \begin{pmatrix} x_{1,1} & \cdots & x_{n,1} \\
\vdots & & \vdots \\
x_{1,m} & \cdots & x_{n,m} \end{pmatrix} \in \mathcal{M}_{m,n}(\mathbb{F}_p)
$$

The rank weight of $\mathbf{x}$ is defined as the rank of this matrix (which does not depend on the choice of the basis).

$$w_r(\mathbf{x}) = \text{Rank } \mathcal{M}(\mathbf{x}) \in [0, \min(m, n)]$$
Example

Let $\mathbb{F}_8 = \mathbb{F}_{2^3}$ and let $\alpha$ such that $\mathbb{F}_8 \simeq \mathbb{F}_2[\alpha] = \text{Vect}(1, \alpha, \alpha^2)$.

Example

$$x = (1, \alpha, \alpha^2 + 1, \alpha + 1) \in \mathbb{F}_8^4$$

$$\mathcal{M}(x) = \begin{pmatrix}
1 & 0 & 1 & 1 \\
0 & 1 & 0 & 1 \\
0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0
\end{pmatrix}$$

$$w_r(x) = 3$$
Support in rank metric

Definition (Rank support)

The support of a word $\mathbf{x} = (x_1, \ldots, x_n) \in (\mathbb{F}_{q^m})^n$ is the subspace of $\mathbb{F}_{q^m}$ generated by its coordinates:

$$\text{Supp}(\mathbf{x}) = \langle x_1, \ldots, x_n \rangle_{\mathbb{F}_q} \subseteq \mathbb{F}_{q^m}$$

Hamming metric: $w_h(\mathbf{x}) = |\text{Supp}(\mathbf{x})|$

Rank metric: $w_r(\mathbf{x}) = \text{dim}(\text{Supp}(\mathbf{x}))$
To reduce the memory footprint of a generator matrix, we define ideal codes.

**Definition (Double circulant code)**

A double circulant code is a code $C[2n, n]$ which admits a double circulating matrix as a generating matrix:

$$G = \begin{pmatrix} a_0 & a_1 & \ldots & a_{n-1} & & b_0 & b_1 & \ldots & b_{n-1} \\ a_{n-1} & a_0 & \ldots & a_{n-2} & & b_{n-1} & b_0 & \ldots & b_{n-2} \\ \vdots & \ddots & \ddots & \vdots & & \vdots & \ddots & \ddots & \vdots \\ a_1 & a_2 & \ldots & a_0 & & b_1 & b_2 & \ldots & b_0 \end{pmatrix}$$
**Definition (Ideal matrix)**

Let $P(X)$ a polynomial in $\mathbb{F}_q[X]$ of degree $n$. A square matrix $M$ of size $n \times n$ is ideal modulo $P$ generated by $f(X)$ when it is of the form:

$$M = \begin{pmatrix}
    f(X) \mod P \\
    Xf(X) \mod P \\
    \vdots \\
    X^{n-1}f(X) \mod P
\end{pmatrix}.$$ 

**Definition (Ideal code)**

An ideal code is a code $C[2n, n]$ having $G = (G_1 | G_2)$ as a generator matrix where $G_1$ and $G_2$ are two ideal matrices.
Difficult problems in rank metric

Definition (Rank Syndrome Decoding $\text{RSD}(n, k, w)$)

Given a random parity check matrix $H \in M_{n-k, n}(\mathbb{F}_q)$ and a syndrome $s = He$ for $e$ an error of rank weight $w(e) = w$, find $e$. 
Difficult problems in rank metric

**Definition (Rank Syndrome Decoding RSD(n, k, w))**

Given a random parity check matrix $H \in \mathcal{M}_{n-k,n}(\mathbb{F}_q)$ and a syndrome $s = He$ for $e$ an error of rank weight $w(e) = w$, find $e$.

**Definition (Rank Support Learning RSL(n, k, w, ℓ))**

Given a random parity check matrix $H \in \mathcal{M}_{n-k,n}(\mathbb{F}_q)$ and $\ell$ syndromes $s_i = He_i$ for $e_i$ errors of same support $E$ a subspace of dimension $w$, find $E$. 
Difficult problems in rank metric

Definition (Ideal Rank Syndrome Decoding IRSD($n, k, w$))

Given an ideal random parity check matrix $H \in \mathcal{M}_{n-k,n}(\mathbb{F}_q)$ and a syndrome $s = He$ for $e$ an error of rank weight $w(e) = w$, find $e$.

Problematic with the structure:
- Quantum attacks
- Potential weaknesses

1. Ronald Cramer, Léo Ducas et Benjamin Wesolowski. “Mildly short vectors in cyclotomic ideal lattices in quantum polynomial time”. In: Journal of the ACM (JACM) 68.2 (2021), p. 1–26.
An LRPC code is a code which admits a parity check matrix whose coordinates belong to a subspace of $\mathbb{F}_{q^m}$ of small dimension.

**Definition (LRPC codes)**

Let $H = (h_{ij})_{1 \leq i \leq n-k, 1 \leq j \leq n} \in \mathbb{F}_{q^m}^{(n-k) \times n}$ be a full-rank matrix such that its coordinates generate an $\mathbb{F}_q$-subspace $F$ of small dimension $d$:

$$F = \langle h_{ij} \rangle_{\mathbb{F}_q}.$$

Let $C$ be the code with parity-check matrix $H$. By definition, $C$ is an $[n, k]$ LRPC code of dual weight $d$. Such a matrix $H$ is called a homogeneous matrix of weight $d$ and support $F$. 

Low Rank Parity Check Codes
Example

Let us consider again the field $\mathbb{F}_8 = \text{Vect}(1, \alpha, \alpha^2)$

$H = \begin{pmatrix}
1 & \alpha & \alpha \\
\alpha & 0 & \alpha + 1 \\
\alpha & \alpha & \alpha \\
\end{pmatrix}$

is of rank 3 as an $\mathbb{F}_{q^m}$-matrix but the $\mathbb{F}_q$-subspace generated by its coordinates is of dimension 2.

$(1, \alpha, \alpha, 0, \alpha + 1, \alpha, \alpha, \alpha) \rightarrow \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{pmatrix}$
LRPC decoding

Problem

Let $E = \langle e_1, \ldots, e_r \rangle$ an (unknown) subspace of $\mathbb{F}_{q^m}$ of dimension $r$ and $F = \langle f_1, \ldots, f_d \rangle$ a (given) subspace of $\mathbb{F}_{q^m}$ of dimension $d$.

Given an LRPC matrix $H \in F^{n-k \times n}$ and $s = He$ where $e \in E^n$, find $E$. 
LRPC decoding

Problem

Let $E = \langle e_1, ..., e_r \rangle$ an (unknown) subspace of $\mathbb{F}_{q^m}$ of dimension $r$ and $F = \langle f_1, ..., f_d \rangle$ a (given) subspace of $\mathbb{F}_{q^m}$ of dimension $d$. Given an LRPC matrix $H \in F^{n-k \times n}$ and $s = He$ where $e \in E^n$, find $E$.

The coordinates of $s$ belong to the product space $EF = \text{Vect}\{ef \mid e \in E, f \in F\} = \langle e_1 f_1, ..., e_r f_1, ..., e_1 f_d, ..., e_r f_d \rangle$. 
**Problem**

Let \( E = \langle e_1, \ldots, e_r \rangle \) an (unknown) subspace of \( \mathbb{F}_{q^m} \) of dimension \( r \) and \( F = \langle f_1, \ldots, f_d \rangle \) a (given) subspace of \( \mathbb{F}_{q^m} \) of dimension \( d \).

Given an LRPC matrix \( H \in \mathbb{F}_{q^m}^{n-k \times n} \) and \( s = He \) where \( e \in E^n \), find \( E \).

The coordinates of \( s \) belong to the product space

\[
EF = \text{Vect}\{ ef | e \in E, f \in F \} = \langle e_1 f_1, \ldots, e_r f_1, \ldots, e_1 f_d, \ldots, e_r f_d \rangle.
\]

The subspaces \( f_i^{-1} EF \) all contain \( E \) since for example

\[
f_1^{-1} EF = \langle e_1, \ldots, e_r, \ldots, f_1^{-1} e_1 f_d, \ldots, f_1^{-1} e_r f_d \rangle.
\]

So one can hope:

\[
\bigcap_{i=1}^{d} f_i^{-1} EF = E
\]
Algorithm 1: Rank Support Recovery (RSR) algorithm

**Data:** $F = \langle f_1, \ldots, f_d \rangle$ an $\mathbb{F}_q$-subspace of $\mathbb{F}_{q^m}$,

$s = (s_1, \cdots, s_{n-k}) \in \mathbb{F}_{q^m}^{(n-k)}$ a syndrome of an error $e$ of weight $r$ and of support $E$

**Result:** A candidate for the vector space $E$

//Part 1: Compute the vector space $EF$

1. Compute $S = \langle s_1, \cdots, s_{n-k} \rangle$

//Part 2: Recover the vector space $E$

2. $E \leftarrow \bigcap_{i=1}^{d} f_i^{-1} S$ **return** $E$
Failure probability

Two possible cases of failure:

- \( S \subsetneq EF \), the coordinates of the syndrome do not generate the entire space \( EF \), or
Failure probability

Two possible cases of failure:

- $S \subsetneq EF$, the coordinates of the syndrome do not generate the entire space $EF$, or

- $E \subsetneq S_1 \cap \cdots \cap S_d$, the chain of intersection generates a subspace strictly bigger than $E$. 





Failure probability

Two possible cases of failure:

- $S \subsetneq EF$, the coordinates of the syndrome do not generate the entire space $EF$, or
- $E \subsetneq S_1 \cap \cdots \cap S_d$, the chain of intersection generates a subspace strictly bigger than $E$.

**Proposition**

The Decoding Failure Rate of algorithm RSR is bounded from above by:

$$q^{rd-(n-k)-1} + q^{-(d-1)(m-rd-r)}$$
Application of LRPC to cryptography

Definition (Key generation)

Let \( U = (A|B) \) an LRPC matrix of weight \( d \).

\[
\begin{align*}
    pk &= H = (I | A^{-1}B) \\
    sk &= U
\end{align*}
\]
Application of LRPC to cryptography

**Definition (Key generation)**

Let \( U = (A|B) \) an LRPC matrix of weight \( d \).

\[
\begin{align*}
\text{pk} & = H = (I|A^{-1}B) \\
\text{sk} & = U
\end{align*}
\]

**Definition (Encaps)**

Choose an error support \( E \) of dimension \( r \). Pick a random error \( e \) in \( E^n \) and send ciphertext \( c = He \). The shared secret is \( \text{Hash}(E) \).
Application of LRPC to cryptography

Definition (Key generation)
Let $U = (A|B)$ an LRPC matrix of weight $d$.

$$\begin{align*}
pk &= H = (I|A^{-1}B) \\
sk &= U
\end{align*}$$

Definition (Encaps)
Choose an error support $E$ of dimension $r$. Pick a random error $e$ in $E^n$ and send ciphertext $c = He$. The shared secret is $\text{Hash}(E)$.

Definition (Decaps)
Compute $s = Ae = Ue$ and use LRPC decoding to find $E$. 
### ROLLO-II parameters

| Instance     | $q$ | $n$ | $m$ | $r$ | $d$ | Security | DFR    |
|--------------|-----|-----|-----|-----|-----|----------|--------|
| ROLLO-II-128 | 2   | 189 | 83  | 7   | 8   | 128      | $2^{-134}$ |
| ROLLO-II-192 | 2   | 193 | 97  | 8   | 8   | 192      | $2^{-130}$ |
| ROLLO-II-256 | 2   | 211 | 97  | 8   | 9   | 256      | $2^{-136}$ |

**Figure**: Parameters for ROLLO-II.

| Instance     | pk size | sk size | ct size | Security |
|--------------|---------|---------|---------|----------|
| ROLLO-II-128 | 1941    | 40      | 2089    | 128      |
| ROLLO-II-192 | 2341    | 40      | 2469    | 192      |
| ROLLO-II-256 | 2559    | 40      | 2687    | 256      |

**Figure**: Resulting sizes in bytes for ROLLO-II using NIST seed expander initialized with 40 bytes long seeds. The security is expressed in bits.
**ROLLO-I parameters**

| Instance   | $q$ | $n$ | $m$ | $r$ | $d$ | Security | DFR     |
|------------|-----|-----|-----|-----|-----|----------|---------|
| ROLLO-I-128 | 2   | 83  | 67  | 7   | 8   | 128      | $2^{-28}$ |
| ROLLO-I-192 | 2   | 97  | 79  | 8   | 8   | 192      | $2^{-34}$ |
| ROLLO-I-256 | 2   | 113 | 97  | 9   | 9   | 256      | $2^{-33}$ |

**Figure:** Parameters for ROLLO-I.

| Instance   | pk size | sk size | ct size | Security |
|------------|---------|---------|---------|----------|
| ROLLO-I-128 | 696     | 40      | 696     | 128      |
| ROLLO-I-192 | 958     | 40      | 958     | 192      |
| ROLLO-I-256 | 1371    | 40      | 1371    | 256      |

**Figure:** Resulting sizes in bytes for ROLLO-I using NIST seed expander initialized with 40 bytes long seeds. The security is expressed in bits.
### Summary

1. Background on rank metric and LRPC codes
2. Presentation of LRPC-MS
3. Analysis of the decoding failure rate
4. Bonus: advances in LRPC implementations
5. Conclusion and perspectives
Idea

**Definition (Key generation)**

Let \( U = (A | B) \) an LRPC matrix of weight \( d \).

\[
\begin{align*}
    pk &= H = (I | A^{-1}B) \\
    sk &= U
\end{align*}
\]

**Definition (Encaps)**

Choose an error support \( E \) of dimension \( r \). Pick \( \ell \) random errors \( e_i \) in \( E^n \) for \( 1 \leq i \leq \ell \) and send ciphertexts \( c_i = H e_i \). The shared secret is \( \text{Hash}(E) \).

**Definition (Decaps)**

Compute \( s_i = A c_i = U e_i \) and use LRPC decoding with multiple syndromes to find \( E \).
The LRPC decoding algorithm has several syndromes as inputs

\[ s_i = Ue_i. \]

\[ S = UV \]
Algorithm 2: Rank Support Recovery (RSR) algorithm with multiple syndromes

**Data:** $F = \langle f_1, ..., f_d \rangle$ an $\mathbb{F}_q$-subspace of $\mathbb{F}_q^m$, $S = (s_{ij}) \in \mathbb{F}_q^{(n-k) \times \ell}$ the $\ell$ syndromes of error vectors of weight $r$ and support $E$

**Result:** A candidate for the vector space $E$

1. Compute $S = \langle s_{11}, \cdots, s_{(n-k)\ell} \rangle$

2. $E \leftarrow \bigcap_{i=1}^{d} f_i^{-1}S$

3. return $E$
For $k \geq \ell$ and for $U$ and $V$ random variables chosen uniformly in $F^{(n-k)\times n}$ and $E^{n\times \ell}$ respectively, the Decoding Failure Rate of algorithm $RSR(F, UV, r)$ is bounded from above by:

$$(n - k)q^{rd - (n-k)\ell} + q^{-(d-1)(m-rd-r)}$$
Parameters with an ideal structure

| Instance     | $q$ | $n$ | $k$ | $m$ | $r$ | $d$ | $\ell$ | Security | DFR    |
|--------------|-----|-----|-----|-----|-----|-----|--------|----------|--------|
| ILRPC-MS-128 | 2   | 94  | 47  | 83  | 7   | 8   | 4      | 128      | $2^{-126}$ |
| ILRPC-MS-192 | 2   | 134 | 67  | 101 | 8   | 8   | 4      | 192      | $2^{-198}$ |

**Figure:** Parameters for ILRPC-MS

| Instance     | pk size | sk size | ct size  | Security |
|--------------|---------|---------|----------|----------|
| ILRPC-MS-128 | 488     | 40      | 1,951    | 128      |
| ILRPC-MS-192 | 846     | 40      | 3,384    | 192      |

**Figure:** Resulting sizes in bytes for ILRPC-MS using NIST seed expander initialized with 40 bytes long seeds. The security is expressed in bits.
## Parameters without an ideal structure

**Figure:** Parameters for LRPC-MS

| Instance     | $q$ | $n$ | $k$ | $m$ | $r$ | $d$ | $\ell$ | Security | DFR   |
|--------------|-----|-----|-----|-----|-----|-----|--------|----------|-------|
| LRPC-MS-128  | 2   | 34  | 17  | 113 | 9   | 10  | 13     | 128      | $2^{-126}$ |
| LRPC-MS-192  | 2   | 42  | 21  | 139 | 10  | 11  | 15     | 192      | $2^{-190}$ |

**Figure:** Resulting sizes in bytes for LRPC-MS using NIST seed expander initialized with 40 bytes long seeds. The security is expressed in bits.

| Instance     | pk size | sk size | ct size   | Security |
|--------------|---------|---------|-----------|----------|
| LRPC-MS-128  | 4,083   | 40      | 3,122     | 128      |
| LRPC-MS-192  | 7,663   | 40      | 5,474     | 192      |
## Comparison to other KEMs

| Instance                       | 128 bits | 192 bits   |
|--------------------------------|----------|------------|
| **LRPC-MS**                    | 7,205    | 12,445     |
| Loong.CCAKEM-III               | 18,522   | N/A        |
| FrodoKEM                       | 19,336   | 31,376     |
| Loidreau cryptosystem          | 36,300   | N/A        |
| Classic McEliece               | 261,248  | 524,348    |

**Figure:** Comparison of sizes of unstructured post-quantum KEMs. The sizes represent the sum of public key and ciphertext expressed in bytes.

| Instance   | 128 bits | 192 bits |
|------------|----------|----------|
| **ILRPC-MS** | 2,439    | 4,230    |
| BIKE       | 3,113    | 6,197    |
| ROLLO-II   | 4,030    | 4,810    |
| HQC        | 6,730    | 13,548   |

**Figure:** Comparison of sizes of structured code-based KEMs. The sizes represent the sum of public key and ciphertext expressed in bytes.
Specificity to rank metric

- Sending errors with the same support does not make sense in Hamming metric
- Additional information given by multiple syndromes can be specifically leveraged by LRPC decoding algorithm
IND-CPA proof

**Definition (LRPC indistinguishability)**

Given a matrix $H \in \mathbb{F}_{qm}^{(n-k) \times k}$, distinguish whether the code $C$ with the parity-check matrix $(I_{n-k}|H)$ is a random code or an LRPC code of weight $d$.

**Definition (Rank Support Learning $\text{RSL}(n, k, w, \ell)$)**

Given a random parity check matrix $H \in \mathcal{M}_{n-k,n}(\mathbb{F}_q)$ and $\ell$ syndromes $s_i = He_i$ for $e_i$ errors of same support $E$ a subspace of dimension $w$, find $E$.

$\Rightarrow$ considered difficult as long as $\ell \leq k(r - 3)$ (without ideal structure) or $\ell \leq r - 3$ (with ideal structure).
Summary

1. Background on rank metric and LRPC codes
2. Presentation of LRPC-MS
3. Analysis of the decoding failure rate
4. Bonus : advances in LRPC implementations
5. Conclusion and perspectives
Objective

We fix $E$ and $F$ subspaces of $\mathbb{F}_{q^m}$ of dimension $r$ and $d$ respectively such that $EF$ is of dimension $rd$. We also impose $q = 2$.

Theorem

For $n_1 + n_2 \leq n$ and for $U$ and $V$ random variables chosen uniformly in $F^{n_1 \times n}$ and $E^{n \times n_2}$ (respectively),

$$\mathbb{P}(\text{Supp}(UV) \neq EF) \leq n_1 q^{rd - n_1 n_2}$$
Product of matrices
Impossible to use Leftover Hash Lemma

**Lemma (Leftover Hash Lemma)**

Let $\{\Phi_r\}_{r \in R}$ be a $(1 + \alpha)/m$-almost universal family of hash functions from $S$ to $T$, where $m := |T|$. Let $H$ and $X$ be independent random variables, where $H$ is uniformly distributed over $R$, and $X$ takes values in $S$. If $\beta$ is the collision probability of $X$, and $\delta$ is the distance of $(H, \Phi_H(X))$ from uniform on $R \times T$, then

$$\delta \leq \frac{1}{2} \sqrt{m\beta + \alpha}.$$
Main proof idea

We fix \( \phi \) a linear form on \( EF \) and we study the probability to have \( \phi(UV) = 0 \).

**Lemma**

We denote \( v \) a column vector in \( E^n \).

For a fixed \( U \), \( \varphi_U : v \mapsto \phi(Uv) \) is a linear map from \( E^n \) to \( \mathbb{F}_q^{n_1} \) so the distribution of \( \phi(Uv) \) is uniform in \( \text{Im}(\varphi_U) \subset \mathbb{F}_q^{n_1} \).

We shall study the rank of \( \varphi_U \).

Indeed, when \( \text{Rank}(\varphi_U) = i \),

\[
\mathbb{P}(\phi(Uv) = 0) = q^{-i}
\]
Main proof idea

We fix $\phi$ a linear form on $EF$ and we study the probability to have $\phi(UV) = 0$.

Lemma

We denote $v$ a column vector in $E^n$.

For a fixed $U$, $\varphi_U : v \mapsto \phi(Uv)$ is a linear map from $E^n$ to $F_q^{n_1}$ so the distribution of $\phi(Uv)$ is uniform in $\text{Im}(\varphi_U) \subset F_q^{n_1}$.

We shall study the rank of $\varphi_U$.

Indeed, when $\text{Rank}(\varphi_U) = i$,

\[ \mathbb{P}(\phi(Uv) = 0) = q^{-i} \]

\[ \mathbb{P}(\phi(UV) = 0) = q^{-in_2} \]
By duality, the linear form $\phi$ is associated to a vector $\tau$ in $EF$ such that $\phi(x) = \langle \tau, x \rangle$. 

$\tau$ can be written:

$$\tau = \sum_{i=1}^{s} e_i f_i$$

where $(e_1, ..., e_s)$ and $(f_1, ..., f_s)$ are linearly independent elements in $E$ and $F$. $s$ is also called the tensor rank of $\tau$.

These tuples can be completed to form basis $(e_1, ..., e_s, ..., e_r)$ and $(f_1, ..., f_s, ..., f_d)$. 

Defining basis
We denote $U_{ij}^{(k)}$ the coordinates of $U_{ij}$ in the basis we chose previously.

\[
\varphi_U((0, \ldots, 0, e_k, 0, \ldots, 0)) = \phi(U(0, \ldots, 0, e_k, 0, \ldots, 0))
\]
\[
= (\phi(e_k U_{ij}))_{1 \leq i \leq n_1}
\]
\[
= (\langle \tau, \sum U_{ij}^{(l)} e_k f_l \rangle)_{1 \leq i \leq n_1}
\]
\[
= \begin{cases} 
(U_{ij}^{(k)})_{1 \leq i \leq n_1} & k \leq s \\
0 & k > s
\end{cases}
\]
So the matrix of $\varphi_U$ looks like

$$
\begin{pmatrix}
* & \cdots & * & 0 & \cdots & 0 \\
* & \cdots & * & 0 & \cdots & 0 \\
& & & & & \\
& & & & & \\
& & & & & \\
& & & & & \\
* & \cdots & * & 0 & \cdots & 0 \\
* & \cdots & * & 0 & \cdots & 0
\end{pmatrix}
$$

where each $*$ is an independent uniform random variable.
End of the proof

The rank \( \varphi_U \) thus follows the law of a random variable \( R_s \).

\[
\mathbb{P}(\text{Supp}(U V) \subset \ker(\phi_\tau)) = \sum_{i=0}^{n_1} \mathbb{P}(\text{Supp}(U V) \subset \ker(\phi_\tau) \mid \text{Rank}(\varphi_U) = i) \\
= \sum_{i=0}^{n_1} q^{-in_2} \mathbb{P}(\text{Rank}(\varphi_U) = i) \\
= \sum_{i=0}^{n_1} q^{-in_2} \mathbb{P}(R_s = i) \\
= \mathbb{E}(q^{-n_2 R_s}) \\
\leq n_1 q^{-n_1 n_2}
\]
Summary

1. Background on rank metric and LRPC codes
2. Presentation of LRPC-MS
3. Analysis of the decoding failure rate
4. Bonus: advances in LRPC implementations
5. Conclusion and perspectives
Implementations

- Efficient
- Easy to use
- Isochronous (or constant-time) $\Rightarrow$ no conditional branching on a secret expression
Long computations in LRPC codes cryptography

**Definition (Key generation)**

Let $U = (A|B)$ an LRPC matrix of weight $d$.

$$
\begin{align*}
\text{pk} &= H = (I|A^{-1}B) \\
\text{sk} &= U
\end{align*}
$$
Long computations in LRPC codes cryptography

Definition (Key generation)

Let \( U = (x|y) \) an ideal LRPC matrix of weight \( d \).

\[
\begin{align*}
    pk &= H = (I|x^{-1}y) \\
    sk &= U
\end{align*}
\]

Inversion in the field \( \mathbb{K} := \mathbb{F}_{2^m}[X]/(P) \approx \mathbb{F}_{(2^m)^n} \)

\( P \) is an irreducible polynomial of degree \( n \) with coefficients in \( \mathbb{F}_{2^m} \).
Natural inversion algorithm

Find $u, v$ such that $ux + vP = 1$.

| $i$  | quotient $q_i$ | remainder $r_i$ | $u_i$  | $v_i$  |
|------|----------------|-----------------|--------|--------|
| 0    |                | $P$             | 1      | 0      |
| 1    |                | $x$             | 0      | 1      |
| ...  |                |                 |        |        |
| $i$  | $r_{i-2}/r_{i-1}$ | $r_{i-2} - q_i r_{i-1}$ | $u_{i-2} - q_i u_{i-1}$ | $v_{i-2} - q_i v_{i-1}$ |
| ...  |                |                 |        |        |
| $k$  | $q_k$          | $r_k$           | $u_k$  | $v_k$  |
| $k + 1$ | $q_{k+1}$ |               | 0      |        |

**Table:** Extended Euclidean algorithm

$\Rightarrow$ can lead to **cache attacks**
A naive approach

Use Euclidean algorithm with naive isochronous techniques.

- Set the number of iterations to a constant.
- Make euclidean divisions isochronous $\Rightarrow$ slow and difficult to implement.
Itoh-Tsuiji algorithm

Idea\(^2\) : compute \(x^{-1} = (x^r)^{-1} x^{r-1}\) where :

\[
r = 1 + 2^m + 2^{2m} + \ldots + 2^{(n-1)m} = \frac{2^{mn} - 1}{2^m - 1}
\]

It is easy to prove that \(x^r \in \mathbb{F}_{2^m}\).

This reduces the inversion in \(\mathbb{F}_{(2^m)^n}\) to :

- The computation of \(x^{r-1}\) and \(x^r\), which can easily be made isochronous;
- An inversion in the smaller field \(\mathbb{F}_{2^m}\);
- \(n\) multiplications in \(\mathbb{F}_{2^m}\).

---

2. Toshiya Itoh et Shigeo Tsujii. “A fast algorithm for computing multiplicative inverses in GF (2m) using normal bases”. In : Information and computation 78.3 (1988), p. 171–177.
Switching to normal basis

Usually, an element $x \in \mathbb{K} = \mathbb{F}_{(2^m)^n}$ is represented in the power basis $\{1, X, ..., X^{n-1}\}$:

$$x = x_0 + x_1 X + ... + x_{n-1} X^{n-1}$$

with $x_i \in \mathbb{F}_{2^m}$.

But it can be more practical to use a normal basis:

$$x = x_0 \alpha + x_1 \alpha^{2^m} + ... + x_{n-1} \alpha^{2(n-1)m}$$

where $\alpha$ is chosen such that $(\alpha, \alpha^{2^m}, \alpha^{2^{2m}}, ..., \alpha^{2(n-1)m})$ is a basis of $\mathbb{K}$ seen as a $\mathbb{F}_{2^m}$-vector space.
Characteristics of a normal basis

- Easy to perform operation $x \mapsto x^{2^m}$
- Multiplication: very expensive

\[ x \cdot y = \left( \sum_i x_i \alpha^{2^{im}} \right) \cdot \left( \sum_j y_j \alpha^{2^{jm}} \right) \]

\[ = \sum_{i,j} x_i y_j \alpha^{2^{im} + 2^{jm}} \]

\[ = \sum_{i,j,k} x_i y_j t_{i,j,k} \alpha^{2^{ik}} \]

Except if you find an optimal normal basis
**Optimal normal basis**

**Definition (Optimal normal basis)**

A optimal normal basis is a basis \((\alpha, \alpha^{2^m}, \alpha^{2^{2m}}, ..., \alpha^{2^{(n-1)m}})\) such that for all \(i\), \(\alpha \alpha^{2^{im}} = \alpha^{2^{a_i m}} + \alpha^{2^{b_i m}}\).

| Scheme     | \(n\) | ONB? |
|------------|-------|------|
| ROLLO-I-128 | 83    | ✓    |
| ROLLO-I-192 | 97    | ✗    |
| ROLLO-I-256 | 113   | ✓    |
| ROLLO-II-128 | 189  | ✓    |
| ROLLO-II-192 | 193   | ✗    |
| ROLLO-II-256 | 211  | ✗    |

**Table:** Existence of an optimal normal basis depending on the value \(n\) for each ROLLO set of parameters.
Smarter square and multiply

$$r - 1 = 2^m + 2^{2m} + ... + 2^{(n-1)m}$$

Square & Multiply $\Rightarrow n - 1$ multiplications.
We find a way to do $\log(n)$.

$$r - 1 = 2^m \left( \log(n-1) \sum_{i=1}^{\log(n-1)} (2^{m2^i} - 1) 2^m(t \mod 2^i) \right)$$
## Performance results

| Algorithm Type                  | ROLLO-I-128  | ROLLO-I-256  | ROLLO-II-128 |
|--------------------------------|--------------|--------------|--------------|
| Non-isochronous algorithm      | 1,030,500    | 1,702,620    | 4,295,704    |
| Isochronous algorithm          | 11,204,649   |              |              |
| Isochronous algorithm (our work) | 3,514,016    | 5,785,700    | 22,859,614   |

**Table**: Duration of the key generation in CPU cycles

1. Nicolas Aragon et al. *Rank-Based Cryptography Library*. [URL: https://rbc-lib.org/](https://rbc-lib.org/).
2. Carlos Aguilar-Melchor et al. “Constant time algorithms for ROLLO-I-128”. *In: SN Computer Science* 2.5 (2021), p. 1–19.
3. Carlos Aguilar-Melchor et al. “Fast and Secure Key Generation for Low Rank Parity Check Codes Cryptosystems”. *In: 2021 IEEE International Symposium on Information Theory (ISIT)*. IEEE. 2021, p. 1260–1265.
Further refinement of our work

| Non-isochronous algorithm | ROLLO-I-128 | ROLLO-I-256 | ROLLO-II-128 |
|---------------------------|-------------|-------------|--------------|
|                           | 1,030,500   | 1,702,620   | 4,295,704    |

| Isochronous algorithm     | ROLLO-I-128 | ROLLO-I-256 | ROLLO-II-128 |
|---------------------------|-------------|-------------|--------------|
|                           | 11,204,649  |             |              |
| Isochronous algorithm (our work) | 3,514,016 | 5,785,700   | 22,859,614   |
| Isochronous algorithm ¹   | 851,823     | 1,477,519   | 4,663,096    |

**Table:** Duration of the key generation in CPU cycles

1. Tung Chou et Jin-Han Liou. “A Constant-time AVX2 Implementation of a Variant of ROLLO”. In: IACR Transactions on Cryptographic Hardware and Embedded Systems (2022), p. 152–174.
Table 4: Cycle counts for key generation, encapsulation, and decapsulation of the ROLLO-I implementations from [AMAB+21] (the paper did not implement ROLLO-II), our ROLLO+ implementation, and the BIKE implementation from [CCK21].

| instance    | key gen. | encap.  | decap  | level | reference     |
|-------------|----------|---------|--------|-------|---------------|
| ROLLO-I-128 | 11034623 | 984432  | 9775241| 1     | [AMAB+21]     |
|             | 11204649 | 320835  | 9744693|       |               |
| ROLLO+I-128 | 851823   | 30361   | 673666 | 1     | this paper    |
| ROLLO+I-192 | 980860   | 38748   | 878398 | 3     |               |
| ROLLO+I-256 | 1477519  | 55353   | 1635966| 5     |               |
| ROLLO+II-128| 4663096  | 70621   | 876533 | 1     | this paper    |
| ROLLO+II-192| 4058419  | 94138   | 1060271| 3     |               |
| ROLLO+II-256| 4947630  | 90021   | 1497315| 5     |               |
| bike11      | 589625   | 114256  | 1643551| 1     | [CCK21]       |
| bike13      | 1668511  | 267644  | 5128078| 3     |               |
Summary

1. Background on rank metric and LRPC codes
2. Presentation of LRPC-MS
3. Analysis of the decoding failure rate
4. Bonus: advances in LRPC implementations
5. Conclusion and perspectives
Conclusion

- New rank metric based cryptosystem with competitive parameters and no ideal structure
- Probabilistic result on the support of the product of two random matrices
- Additional idea to make $m$ down by 10%
- The approach can generalize to RQC but is less efficient in that case
Thank you for your attention!
An explicit method to build an optimal normal basis of $\mathbb{F}_{(2^m)^n}$ over $\mathbb{F}_{2^m}$.

**Theorem**

Let $n$ be an integer prime to $m$ and such that $2n + 1$ is a prime and assume that either:

1. $2$ is primitive in $\mathbb{Z}_{2n+1}$, or
2. $2n + 1 = 3 \pmod{4}$ and $2$ generates the quadratic residues in $\mathbb{Z}_{2n+1}$.

Then $\alpha = \gamma + \gamma^{-1}$ generates an optimal normal basis of $\mathbb{K}$ over $\mathbb{F}_{2^m}$, where $\gamma$ is a primitive $(2n + 1)$-th root of unity.