A key recovery attack against LRPC using decryption failures

Nicolas Aragon[1,2] and Philippe Gaborit[1]

[1] XLIM-DMI, University of Limoges, 123 Avenue Albert Thomas, 87060 Limoges Cedex, France
[2] This work was partially funded by French DGA

Abstract. In this paper we describe a way to exploit decryption failures in Low Rank Parity Check (LRPC) based IND-CPA encryption schemes. In particular we focus on Ideal-LRPC codes.

We first describe an algebraic approach that allows to model the fact that a decryption failure happens.This approach leads to very high complexities but also affects IND-CCA schemes. We then use the fact that an attacker can manipulate the errors that are sent to the decryption oracle in the case of IND-CPA schemes to propose an attack in the spirit of the reaction attack targetting QC-MDPC codes by Guo, Johansson and Stankovski, adapted to the rank metric. We provide examples of parameters that are broken by this attack in an attack model where the number of calls to the decryption oracle is not limited to 2^{64}.

Keywords: Rank metric, Low Rank Parity Check codes, Reaction attack

1 Introduction

Recently code-based cryptography has received a lot of attention, especially since the start of the NIST post-quantum standardization process. Instances of the McEliece cryptosystem in the Hamming metric, using Goppa or MDPC codes, have the disadvantage of having relatively large public keys. A solution to solve this problem is change the metric and use the rank metric, as in the LAKE [1] and LOCKER [2] proposals.

These proposals are based on the Low Rank Parity Check (LRPC) codes [5], which are good candidates for the McEliece cryptosystem because of their low algebraic structure. While the combinatorial attacks ([6], [4]) and the algebraic attacks have already been studied, there is currently no attack exploiting the fact that the decoding algorithm of LRPC codes is probabilistic and can therefore lead to decryption failures.

In this paper we propose a new attack exploiting these decryption failures, in the spirit of [7] targeting the QC-MDPC codes [9] in the Hamming metric.

2 Generalities on the rank metric

We use the same definitions the rank metric and rank metric codes available in LAKE [1] and LOCKER [2] as well as in [3].

In particular, in the following, we will consider $\mathbb{F}_{q^m}$-linear codes embedded with the rank metric. The rank weight of a vector $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{F}_{q^m}^n$ is denoted $\|\boldsymbol{x}\|$.

We recall the notion of support of a word :

Definition 1 (Support of a word). *Let $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{F}_{q^m}^n$. The support E of $\boldsymbol{x}$, denoted* $\mathrm{Supp}(\boldsymbol{x})$*, is the $\mathbb{F}_q$-subspace of $\mathbb{F}_{q^m}$ generated by the coordinates of $\boldsymbol{x}$:*

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

and we have $\dim E = \|\boldsymbol{x}\|$.

The codes used in the following section are ideal codes. We recall the definition here :

Definition 2 (Ideal codes). *Let $P(X) \in \mathbb{F}_q[X]$ be a polynomial of degree n and $\boldsymbol{g}_1, \boldsymbol{g}_2 \in \mathbb{F}_{q^m}^n$. Let $G_1(X) = \sum_{i=0}^{n-1} g_{1i} X^i$ and $G_2(X) = \sum_{j=0}^{n-1} g_{2j} X^j$ the polynomials associated respectively to $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$.*

By definition, the $[2n, n]_{q^m}$ ideal code $\mathcal{C}$ of generator $(\boldsymbol{g}_1, \boldsymbol{g}_2)$ is the code with generator matrix

$$\boldsymbol{G} = \left(\begin{array}{cc|cc} G_1(X) & \bmod P & G_2(X) & \bmod P \\ XG_1(X) & \bmod P & XG_2(X) & \bmod P \\ \multicolumn{2}{c|}{\vdots} & \multicolumn{2}{c}{\vdots} \\ X^{n-1}G_1(X) & \bmod P & X^{n-1}G_2(X) & \bmod P \end{array}\right)$$

More concisely, we have $\mathcal{C} = \{(\boldsymbol{x}\boldsymbol{g}_1 \mod P, \boldsymbol{x}\boldsymbol{g}_2 \mod P), \boldsymbol{x} \in \mathbb{F}_{q^m}^n\}$. We will often omit to precise the polynomial P if there is no ambiguity.

If $\boldsymbol{g}_1$ is invertible, under systematic form, $\mathcal{C} = \{(\boldsymbol{x}, \boldsymbol{x}\boldsymbol{g}), \boldsymbol{x} \in \mathbb{F}_{q^m}^n\}$ with $\boldsymbol{g} = \boldsymbol{g}_1^{-1}\boldsymbol{g}_2 \mod P$.

3 Ideal-LRPC encryption scheme

3.1 High level overview

The LRPC cryptosystem described in [5] is an instantiation of the McEliece setting using the LRPC codes. The most recent use of the ideal LRPC codes as an encryption scheme is described in LOCKER [2]. We recall the key exchange version in figure 1.

This scheme can easily be converted into an encryption scheme by encoding the message m into the error vector $(\boldsymbol{e}_1, \boldsymbol{e}_2)$.

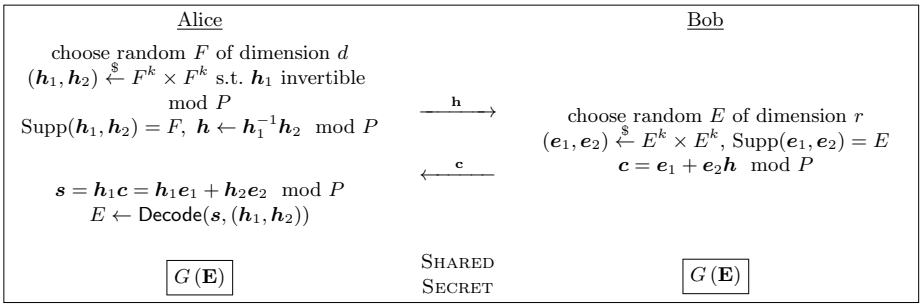

Fig. 1. Key exchange version of the Ideal-LRPC based cryptosystem

3.2 Decoding algorithm

Definition 3. *Product space*

The product space two vector space of U and V, of respective dimensions d_1 and d_2, denoted $< U.V >$ is the vector space spanned by $\{u_1.v_1, \ldots, u_1.v_{d_2}, u_2.v_1, \ldots, u_{d_1}.v_{d_2}\}$.

Let $\{F_1, \ldots F_d\}$ be a basis of F, the support of the secret parity-check matrix, and $\{E_1, \ldots, E_r\}$ be a basis of E, the support of the error vector.

We recall the basic decoding algorithm of LRPC codes from [5] :

Algorithm 1: Basic decoding algorithm of LRPC codes

Data: The syndrome $\boldsymbol{s}$, the low rank parity-check matrix $\boldsymbol{H}$
Result: The error vector $\boldsymbol{e}$

1 Compute the syndrome space $S = \langle \boldsymbol{s}_1, ..., \boldsymbol{s}_{n-k} \rangle$
2 Define $S_i = F_i^{-1}.S$
3 Compute $S_1 \cap ... \cap S_d$. With a high probability, $S_1 \cap ... \cap S_d = E$
4 Solve the system $\boldsymbol{H}\boldsymbol{e}^t = \boldsymbol{s}$ by writing $e_i = \sum_{i=1}^{n} e_{ij}E_j$

Theorem 1. *The probability of failure of algorithm 1 is $q^{-(n-k+1-rd)}$, the probability that the coordinates of the syndrome do not span the whole product space $\langle E.F \rangle$.*

4 A key recovery attack using decryption failures based on algebraic equations

In this section we describe a new attack against the schemes based on the LRPC codes exploiting decryption failures. We consider Ideal-LRPC codes, in particular we assume that the secret parity check matrix $\boldsymbol{H}$ is derived from its first row $(\boldsymbol{h}_1, \boldsymbol{h}_2)$, denoted $\boldsymbol{h}$ in the following.

We know from theorem 1 that if a decryption failure occurs, the vector space spanned by the syndrome coordinates is a subspace of the product space $\langle E.F \rangle$ (more details are given about this in [5]).

The syndrome not spanning the whole product space $\langle E.F \rangle$ is equivalent to its associated matrix not being of rank rd, which can be put in equations by using the fact that all of its $rd \times rd$ minors are null.

Formal product

Even if we never have access to the syndrome associated with the secret matrix when attacking these schemes, we can use a formal product (that was described in [5] in order to reduce the decoding complexity) to write the coordinates of the syndrome in a formal basis of $\langle E.F \rangle$. This gives us an $(n-k) \times rd$ matrix over $\mathbb{F}_q$ that only depends of the coordinates of the secret vector $\boldsymbol{h}$ written in a basis of F, and of the coordinates of the error written in a basis of E. For example if we consider the first coordinate of the syndrome $s_1 = \sum_{j=1}^{n} h_j e_j$, then by unfolding h_j in a basis $\{F_1, \ldots, F_d\}$ of F and e_j in a basis $\{E_1, \ldots, E_r\}$ of E, we can write $s_1 = \sum_{j=1}^{n} \sum_{u=1}^{d} \sum_{v=1}^{r} h_{ju} F_u e_{jv} E_v$. We can view this relation as a linear system with nd unknowns (the h_{ju}) and rd equations in the base field $\mathbb{F}_q$:

$$\begin{pmatrix}
h_{11} & 0 & 0 & h_{21} & 0 & 0 & & h_{n1} & 0 & 0 \\
0 & \ddots & 0 & 0 & \ddots & 0 & \cdots & 0 & \ddots & 0 \\
0 & 0 & h_{11} & 0 & 0 & h_{21} & & 0 & 0 & h_{n1} \\
h_{12} & 0 & 0 & h_{22} & 0 & 0 & & h_{n2} & 0 & 0 \\
0 & \ddots & 0 & 0 & \ddots & 0 & \cdots & 0 & \ddots & 0 \\
0 & 0 & h_{12} & 0 & 0 & h_{22} & & 0 & 0 & h_{n2} \\
 & \vdots & & & \vdots & & & & \vdots & \\
h_{1d} & 0 & 0 & h_{2d} & 0 & 0 & & h_{nd} & 0 & 0 \\
0 & \ddots & 0 & 0 & \ddots & 0 & \cdots & 0 & \ddots & 0 \\
0 & 0 & h_{1d} & 0 & 0 & h_{2d} & & 0 & 0 & h_{nd}
\end{pmatrix}
\begin{pmatrix}
e_{11} \\ e_{12} \\ \vdots \\ e_{1r} \\ e_{21} \\ \vdots \\ e_{2r} \\ \vdots \\ e_{n1} \\ \vdots \\ e_{nr}
\end{pmatrix}
=
\begin{pmatrix}
s_{111} \\ \vdots \\ s_{1d1} \\ s_{112} \\ \vdots \\ s_{1d2} \\ \vdots \\ s_{11r} \\ \vdots \\ s_{1dr}
\end{pmatrix} \tag{1}$$

By doing that for each of the syndrome coordinates s_i, we obtain an $(n-k) \times rd$ matrix in $\mathbb{F}_q$. In our case the number of unknowns is nd since we are considering ideal codes. In the general case, the number of unknowns would grow to $nd(n-k)$.

Complexity and queries analysis

We can try to solve a system in those nd unknowns. We do not give details in this extended abstract, by since each decryption failures allows to obtain $\binom{n-k}{rd}$ equations of degree rd, the complexity of solving this system by linearization

is $((nd)^{rd})^{\omega}$, where ω is the linear algebra constant. The number of queries to the decryption oracle is $\frac{(nd)^{rd}}{\binom{n-k}{rd}} \times \frac{1}{DFR}$ where DFR is the decryption failure rate. Here is an example on a parameter set from [5] :

n	k	q	r	d	DFR	Security parameter	Complexity	Oracle calls
94	47	2	5	5	2^{-23}	128	2^{525}	2^{201}

5 A more efficient approach based on the form of the error

5.1 Attack overview

In this section, we consider an attack model in which the attacker can manipulate the errors sent to the decryption oracle (the case of IND-CPA schemes).

The idea is as follows : if we consider an error vector consisting of l non-null coordinates followed by zeros, the equation 1 becomes $s_1 = \sum_{j=1}^{l} h_j e_j = \sum_{j=1}^{l} \sum_{u=1}^{d} \sum_{v=1}^{r} h_{ju} F_u e_{jv} E_v$.

By making guesses on parts of the secret key, we can find error vectors such that they set coordinates of the syndrome to 0, hence increasing the DFR. The framework of our attack is described figure 2.

1. Choose a subset of l coordinates ($\in \mathbb{F}_q$) in the secret key unfolded in a basis of F
2. Enumerate every possibility for these l coordinates
3. For each guess :
   (a) Compute every error vector such that a subset of the coordinates of s_1 is set to 0
   (b) Send all of these errors to the decryption oracle, and count the number of failures n_{fail}
4. The highest value of n_{fail} corresponds to the correct guess of the secret key
5. Repeat steps 2 to 5 until all coordinates are covered

Fig. 2. Overview of our reaction attack

Then, after recovering the coordinates of $\boldsymbol{h}$, we can recover the secret parity-check matrix $\boldsymbol{H}$:

Proposition 1. *The knowledge of the coordinates of the vector $\boldsymbol{h}$ written in a basis of its support F allows to recover the whole secret parity-check matrix $\boldsymbol{H}$.*

Proof. (Sketch)

We know that $\boldsymbol{h}$ is a row of a parity check matrix of the LRPC code, hence $\boldsymbol{G}\boldsymbol{h}^t = 0$, where $\boldsymbol{G}$ is a generator matrix of the code. When both the support

and the coordinates are unknown, this system is bilinear in the F_{jl} (basis of the support) and the h_i (coordinates).

The knowledge of the coordinates leads to a linear system of dm unknowns in $\mathbb{F}_q$ (the F_{jl}) and nm equations.

5.2 Description of the attack

A first idea is to find error vectors such that $s_1 = 0$: if we guess all the h_{ju}, we obtain a system of lr unknowns (the e_{jv}) and rd equations, one for each coordinate of s_1 written in a basis of $\langle E.F \rangle$. It means that in the case $lr > rd$, we are able to forge errors that fix a coordinate of the syndrome.

Proposition 2. *If λ coordinates of $\boldsymbol{s}$ are set to 0, then the decoding failure probability becomes $q^{-(n-k+1-rd)+\lambda}$.*

Proof. As in [5] we evaluate the probability that the syndrome coordinates do not span the whole vector space $\langle E.F \rangle$, which is now $q^{-(n-k+1-\lambda-rd)}$ because of the λ null coordinates. □

If the number of errors sent to the oracle is high enough, we should be able to tell if the DFR follows the probability given in theorem 1 or in proposition 2. If the DFR is higher than excepted, then the corresponding $\boldsymbol{h}$ is a good candidate for the l first coordinates of the secret key.

The problem with this approach is that we need to enumerate ld coordinates of the secret key at a time, which makes the numbers of errors we have to send to the decryption oracle very high in order to distinguish the correct guess from the $q^{ld} - 1$ others.

We now present an improvement of this approach that allows to enumerate only l coordinates at a time, leading to a smaller complexity.

Setting one coordinate of s_1 to 0

The idea is now to enumerate all the possibilities for the coordinates h_{j1}, that is to say the coordinates corresponding to the vector F_1 of the basis of F (this has to be repeated for other coordinates to recover the full secret key). For each of these guesses we can find error vectors such that $s_{111} = 0$.

Decryption failure analysis

In order to exploit the fact that $s_{111} = 0$, we need to know by how much the decryption failure rate is affected by setting a coordinate of the $(n-k) \times rd$ matrix associated to the syndrome to 0.

Proposition 3. *The probability that a random $n \times m$ matrix M, with $n \geqslant m$, is of rank m knowing that one of its coordinates is fixed to 0, is :*

$$\frac{(q^{n-1}-1)\prod_{i=1}^{m-1}(q^n - q^i)}{q^{nm-1}}$$

Proof. We start by counting the number of $n \times m$ matrices of rank m, considering a coordinate is fixed to 0. We have $(q^{n-1}-1)$ possible choices for the column containing the fixed coordinate. Then for each of the remaining columns, we have q^n choices minus every linear combination of the precedent columns. From that we have $(q^{n-1}-1)\prod_{i=1}^{m-1}(q^n-q^i)$ possible matrices.

We then divide by the total number of possible matrices in order to obtain the probability.

□

Proposition 4. *Given the probabilities p_1 and p_2 :*

- *p_1 the probability that a random $(n-k)\times rd$ matrix has rank rd*
- *p_2 the probability that a random $(n-k)\times rd$ matrix has rank rd knowing that one coordinate is fixed to 0*

When $n-k>rd$, we have $p_1 - p_2 = \frac{1}{q^{n-k}} + \mathcal{O}(q^{\frac{rd-2(n-k)}{q-1}})$.

Proof. We have $p_1 = \frac{\prod_{i=0}^{rd-1}(q^{n-k}-q^i)}{q^{(n-k)rd}}$ and $p_2 = \frac{(q^{n-k-1}-1)\prod_{i=1}^{m-1}(q^{n-k}-q^i)}{q^{(n-k)rd-1}}$ from 3, hence :

$$p_1 - p_2 = \frac{\prod_{i=1}^{rd-1}(q^{n-k}-q^i)}{q^{(n-k)rd}}$$

A calculation shows that, when $n-k>rd$:

$$\frac{\prod_{i=1}^{rd-1}(q^{n-k}-q^i)}{q^{(n-k)rd}} = \frac{1}{q^{n-k}} + \mathcal{O}(q^{\frac{rd-2(n-k)}{q-1}})$$

□

We now want to estimate how many non-null coordinates are needed in the error vector in order to obtain enough samples to distinguish between two probabilities of failure.

Proposition 5. *Suppose we have :*

- *S incorrect guesses with probability of failure p_1*
- *1 correct guess with probability of failure $p_2 = p_1 + \epsilon$*

If we denote by N the number of calls to the decryption oracle for each guess, then in order to distinguish with a good probability the correct guess from the S incorrect ones, N must satisfy the following inequality :

$$\frac{1}{S} \geqslant \frac{Np_1}{(\frac{N(p_2-p_1)}{2})^2}$$

Proof. The sequence of calls to the decryption oracle for a fixed guess can be seen as a Bernoulli process with N trials of parameter either p_1 if the guess was incorrect or p_2 if the guess was correct. The number of decryption errors can thus be seen as a binomial distribution. We denote X_1 (respectively X_2) the random variable following the binomial distribution of parameters N and p_1 (respectively p_2).

For any incorrect guess, the expected value $E(X_1)$ is equal to Np_1, and the variance $V(X_1)$ is $Np_1(1-p_1)$, which is very close to $E(X)$ in our case. In the following we consider that $V(X) = E(X)$.

We want to estimate the probability that the distance from the expected value does not exceed $\frac{E(X_2)-E(X_1)}{2}$: this way we should be able to distinguish between the $S-1$ wrong guesses from the correct one.

We use the Chebyshev's inequality :

$$Pr(|X - E(X)| \geqslant a) \leqslant \frac{V(X)}{a^2}$$

By applying this inequality to our values we get :

$$Pr\left(|X1 - E(X_1)| \geqslant \frac{E(X_2) - E(X_1)}{2}\right) \leqslant \frac{Np_1}{(\frac{N(p_2-p_1)}{2})^2}$$

Since we want to distinguish between S different guesses, we need this probability to be lower than $\frac{1}{S}$, hence the result.

□

We tested this formula by trying to distinguish between 2×6 random matrices from the same matrices with a fixed 0 in the first row. We fixed $S = 2^9$ and $N = 2^{17}$. The results are presented figure 3.

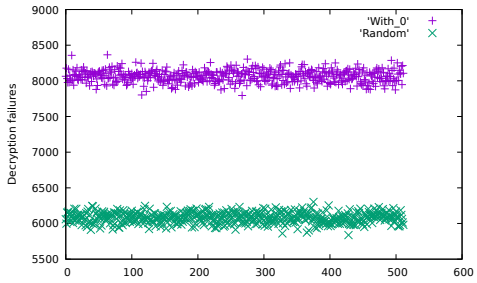

Fig. 3. Results of the distinguisher simulation

5.3 Complexity analysis

To analyze the complexity of this attack, we need to choose a value of l such that the number of queries to the decryption oracle, N, is enough to verify the condition from proposition 5.

When fixing s_{111} to zero we have :

- $S = q^l$, the number of coordinates of $\boldsymbol{h}$ for which we need to enumerate every possibility
- $N = q^{rl-1}$: when setting s_{111} to 0, we need to solve a system consisting of lr unknowns (the coordinates of $\{e_1, \dots, e_l\}$ written in a basis of E) and 1 equation, hence the result
- p_1, the DFR for an incorrect guess, is $q^{-(n-k+1-rd)}$ from theorem 1
- p_2, the DFR for a correct guess, is $p_1 + \frac{1}{q^{n-k}} + \mathcal{O}(q^{\frac{rd-2(n-k)}{q-1}})$ from proposition 4. In our case we will consider that $p_2 = p_1 + \frac{1}{q^{n-k}}$.

Proposition 6. *The total complexity of our attack is :*

$$S \times N \times \lceil \frac{nd}{l} \rceil \times 4r^2d^2m$$

Proof. The total number of queries to the decryption oracle of our reaction attack is :

$$q^l \times q^{rl-1} \times \lceil \frac{nd}{l} \rceil = S \times N \times \lceil \frac{nd}{l} \rceil$$

We know from [5] that the complexity of recovering the support of the error in the decoding algorithm is $4r^2d^2m$: we consider this is the cost of a query to the oracle, hence the result. □

5.4 Scope of the attack

To analyze the impact of our attack on concrete parameters, we consider an attack model where the number of calls to the decryption oracle is not limited.

Impact on parameters Even though the LAKE [1] cryptosystem is not affected by our attack because it uses ephemeral keys, we are going to study the complexity of our attack if these parameters were used in our attack model.

Parameter	n	m	k	q	r	d	l	S	N	DFR	Security parameter	Oracle queries	Attack complexity
LAKE-I-like	94	67	47	2	5	6	17	2^{17}	2^{84}	2^{-30}	128	2^{106}	2^{124}
LAKE-II-like	106	89	53	2	6	7	17	2^{17}	2^{101}	2^{-32}	192	2^{123}	2^{143}
LAKE-III-like	118	107	59	2	6	8	20	2^{20}	2^{119}	2^{-36}	256	2^{145}	2^{164}

As we can see, for these parameters, our attack would reduce the security parameter, but still requires more than 2^{64} calls to the decryption oracle.

However, if we choose other rates than $\frac{1}{2}$ along with a relatively small decryption failure rate, the complexity of this attack can be very small :

n	m	k	q	r	d	l	S	N	DFR	Complexity of [4]	Oracle queries	Attack complexity
87	89	58	2	4	4	6	2^6	2^{23}	2^{-13}	2^{143}	2^{35}	2^{51}

6 Conclusion

In this paper we presented a reaction attack that can be seen as an adaptation of [7] in the rank metric. This attack can break parameters of IND-CPA encryption schemes in an attack model where the number of calls to the decryption oracle is unlimited.

The two proposals based on ideal LRPC submitted to the NIST standardization process are not affected by our attack : LAKE [1] uses ephemeral keys and would require more than 2^{64} queries to achieve the attack, and LOCKER [2] uses the HHK [8] CCA conversion.

Acknowledgements The authors would like to thank Jean-Pierre Tillich for his interesting comments.

References

1. Nicolas Aragon, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Adrien Hauteville, Olivier Ruatta, Jean-Pierre Tillich, and Gilles Zémor. LAKE–Low rAnk parity check codes Key Exchange —-. first round submission to the NIST post-quantum cryptography call, November 2017. *1*, *2*, *9*, *10*
2. Nicolas Aragon, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Adrien Hauteville, Olivier Ruatta, Jean-Pierre Tillich, and Gilles Zémor. LOCKER–LOw rank parity ChecK codes EncRyption –. first round submission to the NIST post-quantum cryptography call, November 2017. *1*, *2*, *10*
3. Nicolas Aragon, Philippe Gaborit, Adrien Hauteville, Olivier Ruatta, and Gilles Zémor. Low rank parity check codes: New decoding algorithms and application to cryptography. 2019. submitted to IEEE Information Theory, preprint available on arXiv. *2*
4. Nicolas Aragon, Philippe Gaborit, Adrien Hauteville, and Jean-Pierre Tillich. A new algorithm for solving the rank syndrome decoding problem. In *Proc. IEEE Int. Symposium Inf. Theory - ISIT*, Vail, USA, 2018. IEEE. *1*, *9*
5. Philippe Gaborit, Gaétan Murat, Olivier Ruatta, and Gilles Zémor. Low rank parity check codes and their application to cryptography. In *Proceedings of the Workshop on Coding and Cryptography WCC'2013*, Bergen, Norway, 2013. Available on `www.selmer.uib.no/WCC2013/pdfs/Gaborit.pdf`. *1*, *2*, *3*, *4*, *5*, *6*, *9*
6. Philippe Gaborit, Olivier Ruatta, and Julien Schrek. On the complexity of the rank syndrome decoding problem. *IEEE Trans. Information Theory*, 62(2):1006–1019, 2016. *1*
7. Qian Guo, Thomas Johansson, and Paul Stankovski. A key recovery attack on MDPC with CCA security using decoding errors. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology - ASIACRYPT 2016*, volume 10031 of *LNCS*, pages 789–815, 2016. *1*, *10*
8. 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. *10*
9. Rafael Misoczki, Jean-Pierre Tillich, Nicolas Sendrier, and Paulo S. L. M. Barreto. MDPC-McEliece: New McEliece variants from moderate density parity-check codes. In *Proc. IEEE Int. Symposium Inf. Theory - ISIT*, pages 2069–2073, 2013. *1*