POST-QUANTUM CRYPTOGRAPHY PROJECT

McNie: Compact McEliece-Niederreiter Cryptosystem

Principal Submitter

This submission is from the following members, listed in alphabetical order:

- Lucky Galvez
- Jon-Lark Kim
- Myeong Jae Kim
- Young-Sik Kim
- Nari Lee

E-mail address: legalvez97@gmail.com, jlkim@sogang.ac.kr, device89@snu.ac.kr, iamyskim@chosun.ac.kr, narilee3@gmail.com

Telephone: +82-10-2046-8836, +82-10-3251-1418

Postal address: Ricci 1401, Sogang University, 35 Baekbeom-ro, Mapo-gu, Seoul 04107, South Korea

College of Electronic and Information 7111, Chosun University, Gwangju, 61452, South Korea

Auxiliary submitters: There are no auxiliary submitters. The principal submitter is the team listed above.

Inventors/developers: The inventors/developers of this submission are the same as the principal submitter. Relevant prior work is credited below where appropriate.

Owner: Same as submitters.

Signature: Lucky Galvez, Jon-Lark Kim, Myeong Jae Kim, Young-Sik Kim, Nari Lee

Contents

Chapter 1

Introduction

As we edge closer to the quantum era, we are confronted not only with obstacles and hurdles, but also with a variety of challenges. The very first challenge we embrace is a change in the current cryptosystem being implemented in almost every section of our lives and businesses. It has arisen from the impact of the Shor's algorithm which effectively break RSA in polynomial time. Consequently, we need to employ a new cryptosystem which is resistant to quantum computing.

There are several cryptosystem candidates named for post-quantum cryptography such as code-based cryptography, lattice-based cryptography, multivariate cryptography and hash-based cryptography. Among these, our interest is the code-based cryptography.

Since the code-based cryptosystem was introduced by McEliece in 1978, it has not been adopted widely until it became one of the candidates for post-quantum cryptography due to its large key size. However, this drawback can be overcome by our suggestion of a hybrid version of McEliece and Niederreiter cryptosystems, called McNie.

McNie provides smaller key sizes employing quasi-cyclicity of matrices for 128-bit, 192-bit and 256-bit securities compared to those of RSA. Note that the shortest key size based on codes for the 128-bit security level is currently 2809 bits which was proposed by Gaborit–Ruatta–Schrek–Tillich–Zémor (2015). McNie provides shorter ones. Especially, the McNie based on a 4-quasi-cyclic $[60, 30]$ LRPC code over $\mathbb{F}_{2^{37}}$ provide a key size of 2775 bits for the 128-bit security. Furthermore, despite of reduction on key sizes, McNie is still secure against structural and ISD attacks by deploying a randomly generated public key. It will be demonstrated in a later section that it is not more difficult to break McEliece than McNie.

Chapter 2

General Algorithm Specification (Part of 2.B.1)

2.1 Parameter space

McNie uses the following parameters : an $(n-k) \times n$ matrix H, an $n \times n$ matrix P, an $(n-k) \times (n-k)$ matrix S, an $l \times n$ matrix G', a block matrix size blk, and an $l \times (n-k)$ matrix F over a finite field $\mathbb{F}_{q^m}$. Here q is a power of a prime and $l > n-k$.

2.2 The McNie public-key Encryption

Alice generates an $(n-k) \times n$ parity check matrix H for an r-error correcting code over a finite field $\mathbb{F}_{q^m}$ belonging to a family of codes with a known efficient decoding algorithm Φ_H. Her secret key consists of H and two other randomly generated matrices with entries in $\mathbb{F}_{q^m}$, an invertible $(n-k) \times (n-k)$ matrix S and an $n \times n$ permutation matrix P.

Alice also randomly generates an $l \times n$ matrix G' with dimension l over $\mathbb{F}_{q^m}$ such that $l > n-k$. She then computes $F = G'P^{-1}H^TS$ which is an $l \times (n-k)$ matrix, and publishes G' and F as her public key.

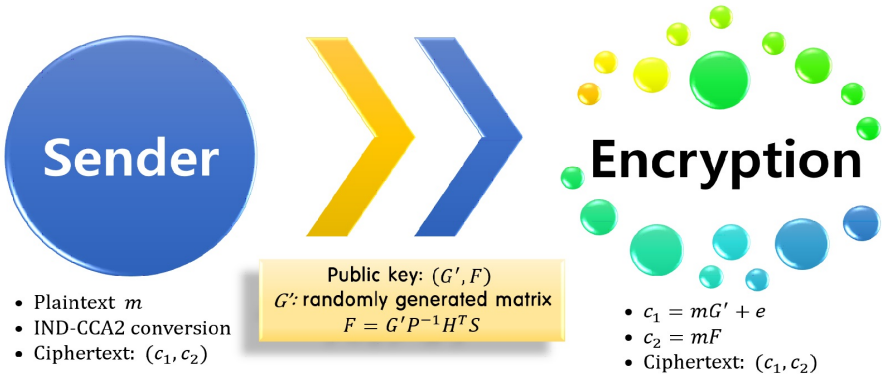
2.3 Encryption

Bob sends a secret message $\mathbf{m}$ to Alice in the form of a length l vector $\mathbf{m}$ over $\mathbb{F}_{q^m}$:

- Generate a random error vector $\mathbf{e}$ of length n over $\mathbb{F}_{q^m}$ of weight at most r which can be decodable by an appropriate decoding algorithm.
- Multiply the vector $\mathbf{m}$ by G' and add the error vector $\mathbf{e}$ to $\mathbf{m}G'$. The resulting vector is $\mathbf{c_1} = \mathbf{m}G' + \mathbf{e}$ of length n.
- Multiply $\mathbf{m}$ by F. The resulting vector is $\mathbf{c_2} = \mathbf{m}F$ of length $n-k$.
- The ciphertext is $\mathbf{c} = (\mathbf{c_1}, \mathbf{c_2})$ of length $2n-k$.

We describe the McNie encryption in Figure 2.1.

Figure 2.1: The McNie encryption



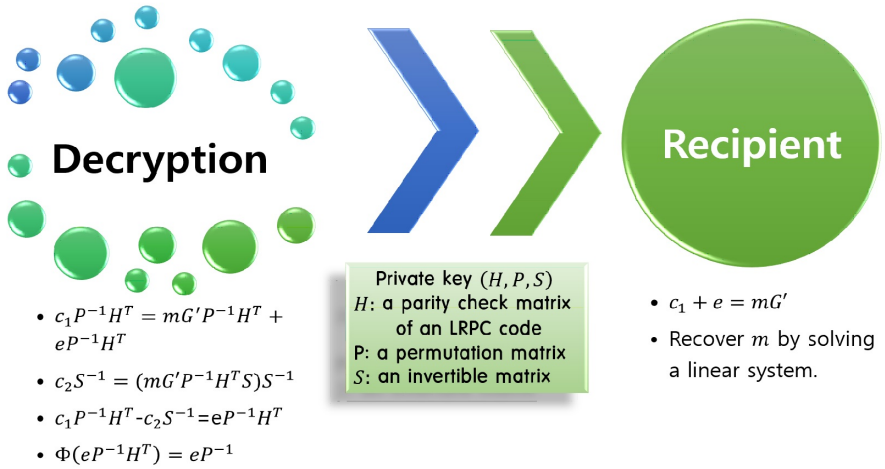
2.4 Decryption

Alice decrypts $\mathbf{c} = (\mathbf{c_1}, \mathbf{c_2})$ by the following steps:

- Compute $\mathbf{s}' = \mathbf{c_1}P^{-1}H^T - \mathbf{c_2}S^{-1} = (\mathbf{m}G' + \mathbf{e})P^{-1}H^T - (\mathbf{m}G'P^{-1}H^TS)S^{-1} = \mathbf{e}P^{-1}H^T$.
- Apply to $\mathbf{s}'$ the efficient decoding algorithm Φ_H to obtain $\mathbf{e}' = \mathbf{e}P^{-1}$.
- Multiply $\mathbf{e}'$ by P to get the error vector $\mathbf{e}$.
- Obtain $\mathbf{m}$ by solving the system $\mathbf{m}G' = \mathbf{c_1} - \mathbf{e}$. In particular, when G' is in standard form, take the first l entries of $\mathbf{c_1} - \mathbf{e}$ to recover $\mathbf{m}$.

We describe the McNie decryption in Figure 2.2.

Figure 2.2: The McNie decryption



2.5 McNie based on LRPC codes

A *Low Rank Parity Check (LRPC)* code of rank d, length n and dimension k over $\mathbb{F}_{q^m}$ is a linear $[n,k]$ block code over $\mathbb{F}_{q^m}$ that has for its parity check matrix an $(n-k)\times n$ matrix $H=(h_{ij})$ such that the sub-vector space of $\mathbb{F}_{q^m}$ generated by its coefficients h_{ij} has dimension at most d. We call this dimension the weight of H. Letting $\mathbb{F}$ be the sub-vector space of $\mathbb{F}_{q^m}$ generated by the coefficients h_{ij} of H, we denote one of its bases by $\{F_1, F_2, \ldots, F_d\}$.

In particular we use quasi-cyclic LRPC codes along with the parameter condition $l > n-k$ in order to have short key sizes. A quasi-cyclic code is an $[n,k]$ linear block code of dimensions $n = mn_0$ and $k = mk_0$ if every cyclic shift of a codeword by n_0 symbols yields another codeword. Any $[n,k]$ Quasi-Cyclic code over $\mathbb{F}_q$ is equivalent to an $[mn_0, mk_0]$ code with an $mk_0 \times mn_0$ generator matrix composed of $m \times m$ circulant matrices,

$$G = \begin{bmatrix} C_{1,1} & C_{1,2} & C_{1,3} & \cdots & C_{1,n_0} \\ C_{2,1} & C_{2,2} & C_{2,3} & \cdots & C_{2,n_0} \\ \vdots & \vdots & \vdots & & \vdots \\ C_{k_0,1} & C_{k_0,2} & C_{k_0,3} & \cdots & C_{k_0,n_0} \end{bmatrix}.$$

A circulant matrix C is uniquely specified by a polynomial formed of the entries of the first row, $c(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_{m-1}x^{m-1}$, i.e., there is a one-to-one mapping between the circulant matrices C_i and the polynomials $c_i(x)$. Note that the sum and product of two circulants is a circulant. In particular, $AB = C$ where $c(x) = a(x)b(x)\ mod\ x^m - 1$.

Specific parameters are given for 3- and 4-quasi-cyclic LRPC codes as follows:

- **2-quasi-cyclic:** In this case, the code length n is even. The parity check matrix H is a double-circulant $\frac{n}{2} \times n$ matrix. Since $l > \frac{n}{2}$, an $l \times n$ matrix G' cannot be a double circulant matrix and it implies that $F = G'H^TS$ is an $l \times \frac{n}{2}$ matrix which cannot be circulant either. So we cannot describe G' and F as a single vector, respectively. This leads to large key sizes for G' and F. We do not consider 2-quasi-cyclic case.

- **3-quasi-cyclic:** Note that n is a multiple of 3 and $l = k = \frac{2n}{3}$. The public keys are $\frac{2n}{3} \times n$ matrix G' and $\frac{2n}{3} \times \frac{n}{3}$ matrix F.

- **4-quasi-cyclic:** Note that n is a multiple of 4, $k = \frac{n}{2}$ and $l = \frac{3n}{4}$. The public keys are $\frac{3n}{4} \times n$ matrix G' and $\frac{3n}{4} \times \frac{n}{2}$ matrix F.

- **s-quasi-cyclic:** When $s \geq 5$, our simulation shows that the key sizes for McNie based on s-quasi cyclic LRPC codes are not shorter than those for McNie based on 3 or 4-quasi cyclic LRPC. So we omit their parameters.

2.5.1 Public key encryption using 3-quasi-cyclic LRPC codes

Key generation

In order to reduce key sizes in McNie, we use circulant matrices and construct quasi-cyclic LRPC codes over $\mathbb{F}_{q^m}$ in the public and private keys. Let n be a multiple of 3 and $blk = \frac{n}{3}$.

- Generate vectors $\mathbf{h}_1$, $\mathbf{h}_2$ and $\mathbf{h}_3$ of length $\frac{n}{3}$ each with entries from an $\mathbb{F}_q$-subspace of $\mathbb{F}_{q^m}$ of small dimension d.
- For $i = 1, 2, 3$, construct the $\frac{n}{3} \times \frac{n}{3}$ circulant matrices H_i whose first row is the vector $\mathbf{h}_i$ and each of the succeeding rows is the cyclic shift of the previous row.
- The $\frac{n}{3} \times n$ matrix $H = \left[\begin{array}{ccc} H_1 & H_2 & H_3 \end{array} \right]$ is a parity check matrix for an LRPC code of weight d. Φ_H is the LRPC decoding algorithm presented in the previous section.

Normally H is stored in vectors $\mathbf{h}_1$, $\mathbf{h}_2$ and $\mathbf{h}_3$. However we are going to store H in a little bit different way. Since H will be used later in the decoding algorithm, we store the $\mathbb{F}_q$-vector $\mathbf{a}_i$'s for $\mathbf{h}_i = \sum_{i=1}^{d} F_i \mathbf{a}_i$ to reduce the decoding time. The following are what we store for H.

- a basis $\{F_1, \ldots, F_d\}$ of F
- the coefficients of $\mathbf{h}_i$'s when they are written in the basis $\{F_1, \ldots, F_d\}$

The coefficients of each $\mathbf{h}_i$ will be stored as a q-ary length n string. Thus the private key size for H is $\frac{d(m+n)}{8} Log(q)$ bytes. The other private key S has a size of $\frac{m(n-k)}{8} Log(q)$ bytes.

- Generate vectors $\mathbf{g}_1$ and $\mathbf{g}_2$ of length $\frac{n}{3}$ each with entries from $\mathbb{F}_{q^m}$.
- As before, construct the $\frac{n}{3} \times \frac{n}{3}$ circulant matrices G_1 and G_2 from $\mathbf{g}_1$ and $\mathbf{g}_2$ respectively, by cyclic shift.
- Define $G' = \left[\begin{array}{ccc} I_{\frac{n}{3}} & 0 & G_1 \\ 0 & I_{\frac{n}{3}} & G_2 \end{array} \right]$.
- Take P to be the $n \times n$ identity matrix.
- Let $S = (H_1 + H_3 G_1^T)^{-1}$, also an $\frac{n}{3} \times \frac{n}{3}$ circulant matrix.
- Compute $F = G' P^{-1} H^T S$, a $\frac{2n}{3} \times \frac{n}{3}$ matrix with the following form :

$$F = \left[\begin{array}{ccc} I_{\frac{n}{3}} & 0 & G_1 \\ 0 & I_{\frac{n}{3}} & G_2 \end{array} \right] \left[\begin{array}{c} H_1^T \\ H_2^T \\ H_3^T \end{array} \right] S = \left[\begin{array}{c} H_1^T + G_1 H_3^T \\ H_2^T + G_2 H_3^T \end{array} \right] S = \left[\begin{array}{c} I_{\frac{n}{3}} \\ F' \end{array} \right],$$

  where $F' = (H_2^T + G_2 H_3^T)(H_1 + H_3 G_1^T)^{-1}$.

- We generate the following keys:

  1. Private keys: (H, S)
     - Key size: $\frac{d(m+n)+m(n-k)}{8} Log(q)$ bytes
       - H: $\frac{d(m+n)}{8} Log(q)$, S: $\frac{m(n-k)}{8} Log(q)$
  2. Public keys: (G', F)
     - Key size: $\frac{nm}{8} Log(q)$ bytes
       - G': $\frac{2nm}{24} Log(q)$, F: $\frac{nm}{24} Log(q)$

There is a probability that the F can not be reduced to column echelon form. In this case, generate new vectors until the reduced column echelon form of F is achieved.

Encryption

The message vector $\bar{\mathbf{m}}$ is over $\mathbb{F}_{q^m}$. The message string $\mathbf{m}$ is a concatenation of the message vector $\bar{\mathbf{m}}$ and a 4-byte string $\mathbf{a}$ which contains the information of the message length, i.e., $\mathbf{m} = (\mathbf{a}||\bar{\mathbf{m}})$. Let α be the number of bytes in 2 blocks. We consider the cases when $s \leq \alpha$ in the following.

Bob sends an s-byte secret message $\mathbf{m}$ to Alice.

- If $s < \alpha$: Define $\mathbf{x}$ as the α-byte string such that $(\mathbf{m}||\mathbf{v})$ where $\mathbf{v}$ is a uniform random $(\alpha - s)$-byte string.
- If $s = \alpha$: Define $\mathbf{x} = \mathbf{m}$.
- Generate an random error vector $\mathbf{e}$ of length n over $\mathbb{F}_{q^m}$ of rank weight at most r which can be decodable by an appropriate decoding algorithm.
- Multiply $\mathbf{x}$ by G' and add the error vector $\mathbf{e}$ to $\mathbf{x}G'$. The resulting vector is $\mathbf{c_1} = \mathbf{x}G' + \mathbf{e}$ of $\frac{nm}{8}$ bytes.
- Multiply $\mathbf{x}$ by F. The resulting vector is $\mathbf{c_2} = \mathbf{x}F$ of $\frac{nm}{24}$ bytes.
- The ciphertext is $\mathbf{c} = (\mathbf{c_1}, \mathbf{c_2})$ of $\frac{nm}{6}$ bytes.

Decryption

Alice decrypts $\mathbf{c} = (\mathbf{c_1}, \mathbf{c_2})$ by the following steps:

- Compute $\mathbf{s}' = \mathbf{c_1}P^{-1}H^T - \mathbf{c_2}S^{-1} = (\mathbf{m}G' + \mathbf{e})P^{-1}H^T - (\mathbf{m}G'P^{-1}H^TS)S^{-1} = \mathbf{e}P^{-1}H^T$.
- Apply the efficient decoding algorithm Φ_H to $\mathbf{s}'$ to obtain $\hat{\mathbf{e}} = \mathbf{e}P^{-1}$.
- Multiply $\hat{\mathbf{e}}$ by P to get the error vector $\mathbf{e}$.
- Compute $\mathbf{x}$ by solving the system $\mathbf{x}G' = \mathbf{c_1} - \mathbf{e}$. In particular, when G' is in standard form, take the first $\frac{lm}{8}$ bytes of $\mathbf{c_1} - \mathbf{e}$ to recover $\mathbf{x}$.
- Get the information of message length from the first 4 bytes of the vector $\mathbf{x}$.
- Obtain the message $\bar{\mathbf{m}}$.

2.5.2 Public key encryption using 4-quasi-cyclic LRPC codes

Key generation

Let n be divisible by 4 and $blk = \frac{n}{4}$.

- Generate vectors $\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_8$ of length $\frac{n}{4}$ each with entries from an $\mathbb{F}_q$-subspace of $\mathbb{F}_{q^m}$ of small dimension d.

- For $i = 1, \ldots, 8$, construct the $\frac{n}{4} \times \frac{n}{4}$ circulant matrices H_i whose first row is the vector $\mathbf{h}_i$ and each of the succeeding rows is the cyclic shift of the previous row.
- The $\frac{n}{2} \times n$ matrix $H = \begin{bmatrix} H_1 & H_2 & H_3 & H_4 \\ H_5 & H_6 & H_7 & H_8 \end{bmatrix}$ is a parity check matrix for an LRPC code of weight d. Φ_H is the LRPC decoding algorithm presented in the previous chapter.

In this case as well, the private key H is stored in the same way as Section 2.2.1.

- a basis $\{F_1, \ldots, F_d\}$ of F
- the coefficients of $\mathbf{h}_i$'s when they are written in the basis $\{F_1, \ldots, F_d\}$

The coefficient of each $\mathbf{h}_i$ will be stored as a q-ary length n string. As a result, the private key size for H and S are $\frac{d(m+2n)}{8} Log(q)$ and $\frac{2m(n-k)}{8} Log(q)$ bytes, respectively.

- Generate vectors $\mathbf{g}_1, \mathbf{g}_2, \mathbf{g}_3$ of length $\frac{n}{4}$ each with entries from $\mathbb{F}_{q^m}$.
- As before, construct the $\frac{n}{4} \times \frac{n}{4}$ circulant matrices G_1, G_2 and G_3 from $\mathbf{g}_1, \mathbf{g}_2$ and $\mathbf{g}_3$, respectively, by cyclic shift.
- Define $G' = \begin{bmatrix} I_{\frac{n}{4}} & 0_{\frac{n}{4}} & 0_{\frac{n}{4}} & G_1 \\ 0_{\frac{n}{4}} & I_{\frac{n}{4}} & 0_{\frac{n}{4}} & G_2 \\ 0_{\frac{n}{4}} & 0_{\frac{n}{4}} & I_{\frac{n}{4}} & G_3 \end{bmatrix}$.
- Take P to be the $n \times n$ identity matrix.
- Generate a nonsingular block-circulant matrix $\bar{S}$ of the form $\bar{S} = \begin{bmatrix} S_1 & S_2 \\ S_3 & S_4 \end{bmatrix}$, where S_1, S_2, S_3, S_4 are $\frac{n}{4} \times \frac{n}{4}$ circulant matrices.
- Compute $\bar{F} = G' P^{-1} H^T S$, a $\frac{3n}{4} \times \frac{n}{2}$ matrix with the following form :

$$\begin{aligned}
\bar{F} &= \begin{bmatrix} I_{\frac{n}{4}} & 0_{\frac{n}{4}} & 0_{\frac{n}{4}} & G_1 \\ 0_{\frac{n}{4}} & I_{\frac{n}{4}} & 0_{\frac{n}{4}} & G_2 \\ 0_{\frac{n}{4}} & 0_{\frac{n}{4}} & I_{\frac{n}{4}} & G_3 \end{bmatrix} \begin{bmatrix} H_1^T & H_5^T \\ H_2^T & H_6^T \\ H_3^T & H_7^T \\ H_4^T & H_8^T \end{bmatrix} \begin{bmatrix} S_1 & S_2 \\ S_3 & S_4 \end{bmatrix} \\
&= \begin{bmatrix} (H_1^T + G_1 H_4^T)S_1 + (H_5^T + G_1 H_8^T)S_3 & (H_1^T + G_1 H_4^T)S_2 + (H_5^T + G_1 H_8^T)S_4 \\ (H_2^T + G_2 H_4^T)S_1 + (H_6^T + G_2 H_8^T)S_3 & (H_2^T + G_2 H_4^T)S_2 + (H_6^T + G_2 H_8^T)S_4 \\ (H_3^T + G_3 H_4^T)S_1 + (H_7^T + G_3 H_8^T)S_3 & (H_3^T + G_3 H_4^T)S_2 + (H_7^T + G_3 H_8^T)S_4 \end{bmatrix} \\
&= \begin{bmatrix} F_1 & F_2 \\ F_3 & F_4 \\ F_5 & F_6 \end{bmatrix}
\end{aligned}$$

- Reduce $\bar{F}$ in column echelon form $F = \bar{F}E = \begin{bmatrix} I_{\frac{n}{4}} & 0_{\frac{n}{4}} \\ 0_{\frac{n}{4}} & I_{\frac{n}{4}} \\ F' & F'' \end{bmatrix}$ using the matrix

$$E = \begin{bmatrix} E_1 & E_2 \\ E_3 & E_4 \end{bmatrix} = \begin{bmatrix} (F_2^{-1}F_1 - F_4^{-1}F_3)^{-1}F_2^{-1} & (F_4^{-1}F_3 - F_2^{-1}F_1)^{-1}F_4^{-1} \\ -F_4^{-1}F_3E_1 & -F_2^{-1}F_1E_2 \end{bmatrix}.$$

(Our experiment shows that there is a very small probability of around 10^{-4} that in the key generation step, the public key F cannot be reduced to column echelon form. In that case, we repeat the key generation step until the matrix F can be reduced to column echelon form.)

- Compute $S = \bar{S}E$.
- We generate the following keys:

  1. Private keys: (H, S)
     - Key size: $\frac{d(m+2n)+2m(n-k)}{8} Log(q)$ bytes
       - H: $\frac{d(m+2n)}{8} Log(q)$, S: $\frac{2m(n-k)}{8} Log(q)$
  2. Public keys: (G', F)
     - Key size: $\frac{nm}{8} Log(q)$ bytes
       - G': $\frac{3nm}{32} Log(q)$, F': $\frac{nm}{16} Log(q)$

There is a probability that F can not be reduced to column echelon form. In this case, generate new vectors until the reduced column echelon form of F is achieved.

Encryption

The message vector $\bar{\mathbf{m}}$ is over $\mathbb{F}_{q^m}$. The message string $\mathbf{m}$ is a concatenation of the message vector $\bar{\mathbf{m}}$ and a 4-byte string $\mathbf{a}$ which contains the information of the message length, i.e., $\mathbf{m} = (\mathbf{a}||\bar{\mathbf{m}})$. Let α be the number of bytes in 3 blocks. We consider the cases when $s \leq \alpha$ in the following.

Bob sends a secret message $\mathbf{m}$ to Alice in s-byte binary vector.

- If $s < \alpha$: Define $\mathbf{x}$ as the α-byte vector such that $(\mathbf{m}||\mathbf{v})$ where $\mathbf{v}$ is a uniform random $(\alpha - s)$-byte vector.
- If $s = \alpha$: Define $\mathbf{x} = \mathbf{m}$.
- Generate an random error vector $\mathbf{e}$ of length n over $\mathbb{F}_{q^m}$ of rank weight at most r which can be decodable by an appropriate decoding algorithm.
- Multiply the vector $\mathbf{x}$ by G' and add the error vector $\mathbf{e}$ to $\mathbf{x}G'$. The resulting vector is $\mathbf{c_1} = \mathbf{x}G' + \mathbf{e}$ of $\frac{nm}{8}$ bytes.
- Multiply $\mathbf{x}$ by F. The resulting vector is $\mathbf{c_2} = \mathbf{x}F$ of length $\frac{nm}{32}$.
- The ciphertext is $\mathbf{c} = (\mathbf{c_1}, \mathbf{c_2})$ of $\frac{5nm}{32}$ bytes.

Decryption

Alice decrypts $\mathbf{c} = (\mathbf{c_1}, \mathbf{c_2})$ by the following steps:

- Compute $\mathbf{s}' = \mathbf{c_1}P^{-1}H^T - \mathbf{c_2}S^{-1} = (\mathbf{m}G' + \mathbf{e})P^{-1}H^T - (\mathbf{m}G'P^{-1}H^TS)S^{-1} = \mathbf{e}P^{-1}H^T$.
- Apply the efficient decoding algorithm Φ_H to $\mathbf{s}'$ to obtain $\hat{\mathbf{e}} = \mathbf{e}P^{-1}$.
- Mutiply $\hat{\mathbf{e}}$ by P to get the error vector $\mathbf{e}$.
- Compute $\mathbf{x}$ by solving the system $\mathbf{x}G' = \mathbf{c_1} - \mathbf{e}$. In particular, when G' is in standard form, take the first $\frac{lm}{8}$ bytes of $\mathbf{c_1} - \mathbf{e}$ to recover $\mathbf{x}$.
- Get the information of message length from the first 4 bytes of the vector $\mathbf{x}$.
- Obtain the message $\bar{\mathbf{m}}$.

2.5.3 Decoding complexity

The original decoding algorithm for LRPC codes is introduced in the Gaborit–Ruatta–Schrek–Tillich–Zémor (GRSTZ) cryptosystem[14]. The decoding complexity in [14] is $r^2(4d^2m + n^2)$. However we modified the original decoding algorithm more efficiently. The following is the modified matrix $K_H^r = (k_{ij})$ which is used in the decoding algorithm reducing the complexity from $r^2(4d^2m + n^2)$ to $r(4d^2mr + n^2)$. The size of K_H^r is an $(n-k)d \times n$ matrix and $k_{i+(n-k)(v-1),j} = h_{ijv}$ for $1 \le v \le d$, $1 \le u \le r$, $1 \le i \le n-k$, and $1 \le j \le n$.

$$K_H^r = \begin{bmatrix} h_{111} & h_{121} & \cdots & h_{1n1} \\ h_{211} & h_{221} & \cdots & h_{2n1} \\ \vdots & \vdots & & \vdots \\ h_{(n-k)11} & h_{(n-k)21} & \cdots & h_{(n-k)n1} \\ h_{112} & h_{122} & \cdots & h_{1n2} \\ h_{212} & h_{222} & \cdots & h_{2n2} \\ \vdots & \vdots & & \vdots \\ h_{(n-k)12} & h_{(n-k)22} & \cdots & h_{(n-k)n2} \\ \\ \vdots & \vdots & & \vdots \\ \\ h_{11d} & h_{12d} & \cdots & h_{1nd} \\ h_{21d} & h_{22d} & \cdots & h_{2nd} \\ \vdots & \vdots & & \vdots \\ h_{(n-k)1d} & h_{(n-k)2d} & \cdots & h_{(n-k)nd} \end{bmatrix} \quad \mathbf{s} = \begin{bmatrix} s_{111} & s_{112} & \cdots & s_{11r} \\ s_{211} & s_{212} & \cdots & s_{21r} \\ \vdots & \vdots & & \vdots \\ s_{(n-k)11} & s_{(n-k)12} & \cdots & s_{(n-k)1r} \\ s_{121} & s_{122} & \cdots & s_{12r} \\ s_{221} & s_{222} & \cdots & s_{22r} \\ \vdots & \vdots & & \vdots \\ s_{(n-k)21} & s_{(n-k)22} & \cdots & s_{(n-k)2r} \\ \\ \vdots & \vdots & & \vdots \\ \\ s_{1d1} & s_{1d2} & \cdots & s_{1dr} \\ s_{2d1} & s_{2d2} & \cdots & s_{2dr} \\ \vdots & \vdots & & \vdots \\ s_{(n-k)d1} & s_{(n-k)d2} & \cdots & s_{(n-k)dr} \end{bmatrix}$$

$$\mathbf{e} = \begin{bmatrix} e_{11} & e_{12} & \cdots & e_{1r} \\ e_{21} & e_{22} & \cdots & e_{2r} \\ \vdots & \vdots & & \vdots \\ e_{n1} & e_{n2} & \cdots & e_{nr} \end{bmatrix}$$

In addition, to minimize failure in the decoding, the above parameters must also satisfy the following relations:

$$\begin{aligned} rd &< n-k \\ r(d+1) &< m \end{aligned}$$

The complexity of the decoding depends on the complexity of the error-correction algorithm Φ_H. In the case of 3-quasi-cyclic and 4-quasi-cyclic LRPC codes, it is $r(4d^2m + n^2)$, the complexity of the modified LRPC decoding algorithm.

Chapter 3

List of Parameter Sets (Part of 2.B.1)

3.0.1 Parameter set `encrypt/3Q_128_1`

PKE with $m = 37$, $blk = 31$, $d = 3$, and $r = 5$

3.0.2 Parameter set `encrypt/3Q_128_2`

PKE with $m = 37$, $blk = 35$, $d = 3$, and $r = 5$

3.0.3 Parameter set `encrypt/3Q_192_1`

PKE with $m = 41$, $blk = 37$, $d = 3$, and $r = 7$

3.0.4 Parameter set `encrypt/3Q_192_2`

PKE with $m = 41$, $blk = 41$, $d = 3$, and $r = 7$

3.0.5 Parameter set `encrypt/3Q_256_1`

PKE with $m = 59$, $blk = 37$, $d = 3$, and $r = 7$

3.0.6 Parameter set `encrypt/3Q_256_2`

PKE with $m = 47$, $blk = 47$, $d = 3$, and $r = 9$

3.0.7 Parameter set `encrypt/4Q_128_1`

PKE with $m = 37$, $blk = 15$, $d = 3$, and $r = 5$

3.0.8 Parameter set `encrypt/4Q_128_2`

PKE with $m = 37$, $blk = 18$, $d = 3$, and $r = 5$

3.0.9 Parameter set `encrypt/4Q_192_1`

PKE with $m = 41$, $blk = 19$, $d = 3$, and $r = 7$

3.0.10 Parameter set `encrypt/4Q_192_2`

PKE with $m = 41$, $blk = 21$, $d = 3$, and $r = 7$

3.0.11 Parameter set `encrypt/4Q_256_1`

PKE with $m = 53$, $blk = 19$, $d = 3$, and $r = 7$

3.0.12 Parameter set `encrypt/4Q_256_2`

PKE with $m = 47$, $blk = 22$, $d = 3$, and $r = 8$

Table 3.1: Suggested parameters using 3-quasi-cyclic LRPC codes

n	k	l	blk	d	r	m	q	Decryption failure 1	Decryption failure 2	Public Key Size (bytes)	Private Key Size (bytes)	Message Size (bytes)	Ciphertext Size (bytes)	Security
93	62	62	31	3	5	37	2	-17	-34	431	194	314	579	128
105	70	70	35	3	5	37	2	-20	-34	486	218	358	653	128
111	74	74	37	3	7	41	2	-17	-26	569	247	454	764	192
123	82	82	41	3	7	41	2	-20	-26	631	274	505	846	192
111	74	74	37	3	7	59	2	-17	-62	819	337	636	1097	256
141	94	94	47	3	9	47	2	-20	-22	829	348	699	1110	256

Table 3.2: Suggested parameters using 3-quasi-cyclic LRPC codes

n	k	l	blk	d	r	m	q	Decryption failure 1	Decryption failure 2	Public Key Size (bytes)	Private Key Size (bytes)	Message Size (bytes)	Ciphertext Size (bytes)	Security
60	30	45	15	3	5	37	2	-16	-34	347	340	215	422	128
72	36	54	18	3	5	37	2	-21	-34	417	401	264	505	128
76	38	57	19	3	7	41	2	-18	-26	487	465	336	590	192
84	42	63	21	3	7	41	2	-21	-26	539	512	373	651	192
76	38	57	19	3	7	53	2	-18	-50	630	584	432	761	256
88	44	66	22	3	8	47	2	-20	-30	647	601	461	781	256

Chapter 4

Design Rationale (Part of 2.B.1)

The GRSTZ cryptosystem based on LRPC codes is the first rank metric based cryptosystem with a random structure that is still considered secure. Moreover, this cryptosystem with quasi-cyclic (QC) LRPC codes provides compact key sizes. This motivates us to design the McNie cryptosystem based on QC-LRPC codes.

The GRSTZ cryptosystem uses a low rank parity check matrix H as the private key and uses the product RG of the generator matrix G of the same code and a masking matrix R as the public key. Known structural attacks on the GRSTZ cryptosystem aim at finding a low weight codeword in the dual of this code to obtain H.

McNie utilizes the decoding algorithm of the GRSTZ cryptosystem and various quasi-cyclic LRPC codes. Therefore, McNie is as secure as the GRSTZ cryptosystem against structural and information set decoding attacks. Furthermore, because McNie employs two different matrices in the private and public keys, that is, the public key G' does not contain any information about the private key H, it can be expected that McNie is harder to break than the GRSTZ cryptosystem.

McNie's encryption scheme is a combination of McEliece and Niedereitter cryptosystem in the sense that $\mathbf{c_1} = \mathbf{m}G' + \mathbf{e}$ resembles the ciphertext in the McEliece cryptosystem while $\mathbf{c_2} = \mathbf{m}F$ resembles the ciphertext in the Niedereitter cryptosystem.

Note that if G is the generator matrix of the code with parity check matrix H and $G' = SGP$, then $F = (SGP)P^{-1}H^TS = 0 \in \mathbb{F}_{q^m}^{l\times(n-k)}$. So $\mathbf{c_2} = 0 \in \mathbb{F}_{q^m}^{n-k}$ and $\mathbf{c_1} = mG' + \mathbf{e} = \mathbf{m}SGP + \mathbf{e}$ is the ciphertext for the McEliece cryptosystem. Hence, it is not more difficult to break the McEliece cryptosystem than our proposed system.

Chapter 5

Detailed Performance Analysis (2.B.2)

5.1 Description of platform

Implementation of McNie was done in C and tested on Intel Core i7-4790 3.60GHz (RAM 8GB, Windows 8).

5.2 Time

In the following tables, we give the time in milliseconds for the key generation, encryption and decryption steps for each parameters using 3-quasi-cyclic and 4-quasi-cyclic LRPC codes.

Table 5.1: Implementation results for McNie using 3-quasi-cyclic LRPC codes

n	k	l	blk	d	r	m	q	security	key gen. (ms)	encryption (ms)	decryption (ms)
93	62	62	31	3	5	37	2	128	62	1.087	1.595
105	70	70	35	3	5	37	2	128	91.5	1.358	2.016
111	74	74	37	3	7	41	2	192	121.8	1.660	2.473
123	82	82	41	3	7	41	2	192	163.5	1.996	2.934
111	74	74	37	3	7	59	2	256	171.1	2.299	3.366
141	94	94	47	3	9	47	2	256	288.5	2.941	4.352

Table 5.2: Implementation results for McNie using 4quasi-cyclic LRPC codes

n	k	l	blk	d	r	m	q	security	key gen. (ms)	encryption (ms)	decryption (ms)
60	30	45	15	3	5	37	2	128	45.9	0.502	1.174
72	36	54	18	3	5	37	2	128	78	0.679	1.608
76	38	57	19	3	7	41	2	192	109	0.824	1.959
84	42	63	21	3	7	41	2	192	142	0.970	2.346
76	38	57	19	3	7	53	2	256	140	1.020	2.466
88	44	66	22	3	8	47	2	256	185.8	1.191	2.873

5.3 Space

Sizes can be computed based on the parameter. Public and private key sizes are given in Tables 3.1 and 3.2. In both the 3-quasi and 4-quasi cyclic cases, the length of the message is $(\frac{lm}{8})$ bytes and the ciphertext is twice the size of the message.

5.4 How parameters affect performance

- Notice that the dimension l of a public key G' should be greater than the dimension $n-k$ of the parity check matrix H. Otherwise, the attacker may recover the message vector from the ciphertext $\mathbf{c_2} = \mathbf{m}F$, where $\mathbf{m} = (m_1, m_2, \ldots, m_l) \in \mathbb{F}_{q^m}^l$ and F an $l \times (n-k)$ matrix. That is, if $l \leq n-k$, the linear system with l unknowns will give us the unique solution, which is the message $\mathbf{m}$. When $l > n-k$, the number of possible solutions of $\mathbf{m}$ in $\mathbf{c_2} = \mathbf{m}F$ is $q^{m(l-n+k)}$.

- Let $H_0 = P^{-1}H^TS$. Then $F = G'P^{-1}H^TS = G'H_0$ and the number of possible H_0 is $q^{m(n-l)(n-k)}$.

5.5 Optimizations

The same as the reference implementation.

Chapter 6

Expected Strength (2.B.4) in General

6.1 Security definitions

The McNie cryptosystem is designed for IND-CCA2 security. See Chapter 8 for the estimates of security of specific parameter sets.

6.2 Rationale

See Chapter 8 for an analysis of semantic and known attacks. This analysis also presents the rationale for these security estimates.

Chapter 7

Expected Strength (2.B.4) for Each Parameter Set

7.1 Parameter set encrypt/random

7.1.1 Parameter set `encrypt/3Q_128_1`

Category 1, classical elementary operations.

7.1.2 Parameter set `encrypt/3Q_128_2`

Category 1, classical elementary operations.

7.1.3 Parameter set `encrypt/3Q_192_1`

Category 3, classical elementary operations.

7.1.4 Parameter set `encrypt/3Q_192_2`

Category 3, classical elementary operations.

7.1.5 Parameter set `encrypt/3Q_256_1`

Category 5, classical elementary operations.

7.1.6 Parameter set `encrypt/3Q_256_2`

Category 5, classical elementary operations.

7.1.7 Parameter set `encrypt/4Q_128_1`

Category 1, classical elementary operations.

7.1.8 Parameter set `encrypt/4Q_128_2`

Category 1, classical elementary operations.

7.1.9 Parameter set `encrypt/4Q_192_1`

Category 3, classical elementary operations.

7.1.10 Parameter set `encrypt/4Q_192_2`

Category 3, classical elementary operations.

7.1.11 Parameter set `encrypt/4Q_256_1`

Category 5, classical elementary operations.

7.1.12 Parameter set `encrypt/4Q_256_2`

Category 5, classical elementary operations.

Chapter 8

Analysis of Known Attacks (2.B.5)

8.1 Semantic security: IND-CCA2 conversion

The security of the McNie cryptosystem relies on the following LRPC problem which can be understood as the codeword existence problem in MDPC code based cryptosystem and the NTRU problem in rank metric sense.

The LRPC problem [14] Given a public matrix G_{pub} it is difficult to recover low weight vector of rank weight d in the dual code.

In McNie, the generator matrix G' and the scrambler matrix S are randomly generated. Using S, the structure of the parity check matrix H of an LRPC code is masked to have F. The approach introduced in [23] for the MDPC cryptosystem on the indistinguishability to random codes and the CCA-2 conversion in [21] can be adapted in McNie as follows:

Notations

$Prep(\mathbf{m})$	:	Preprocessing to a message $\mathbf{m}$, such as data-compression, data-padding and so on. Its inverse is represented as $Prep^{-1}()$.
$Hash(\mathbf{x})$	:	One-way hash function of an arbitrary length binary string x to a fixed length binary string.
$Conv(\bar{\mathbf{z}})$	:	Bijective function which converts a vector $\bar{\mathbf{z}}$ over $\mathbb{F}_{q^m}$ into the corresponding error vector $\mathbf{z}$ of length n with a constant rank weight r. Its inverse is represented as $Conv^{-1}()$.
$Gen(\mathbf{x})$	:	Generator of a cryptographically secure pseudo random sequences of arbitrary length from a fixed length seed x.
$Msb_{x_1}(x_2)$	:	The left x_1 bits of x_2.
$Lsb_{x_1}(x_2)$	:	The right x_1 bits of x_2.
$Const$	:	Predetermined constant used in public.
$Rand$	:	Random source which generates a truly random (or computationally indistinguishable pseudo random) sequence.
$\mathcal{E}^{McNie}(x, z)$		Encryption of x using the McNie PKC with an error vector z.
$\mathcal{D}^{McNie}(x)$	:	Decryption of x using the McNie PKC.

Table 8.1: McNie conversion

Encryption of m:		**Decryption of c:**	
$\mathbf{r}$	$:= Rand$	y_5	$:= Msb_{n-l}(\mathbf{c})$
$\bar{\mathbf{m}}$	$:= Prep(\mathbf{m})$	z	$:= Lsb_{2n}(\mathbf{c})$
y_1	$:= Gen(\mathbf{r}) \oplus (\bar{\mathbf{m}}\|\|Const)$	$\mathbf{c_1}$	$:= Msb_n(z)$
y_2	$:= \mathbf{r} \oplus Hash(y_1)$	$\mathbf{c_2}$	$:= Lsb_n(z)$
$(y_5\|\|y_4\|\|y_3)$	$:= (y_2\|\|y_1)$	y_3	$:= \mathcal{D}^{McNie}(\mathbf{c_1}\|\|\mathbf{c_2})$
$\mathbf{e}$	$:= Conv(y_4)$	$\mathbf{e}$	$:= y_3 G' \oplus \mathbf{c_1}$
$(\mathbf{c_1}\|\|\mathbf{c_2})$	$:= \mathcal{E}^{McNie}(y_3, \mathbf{e})$	y_4	$:= Conv^{-1}(\mathbf{e})$
$\mathbf{c}$	$:= (y_5\|\|\mathbf{c_1}\|\|\mathbf{c_2})$	$(y_2\|\|y_1)$	$:= (y_5\|\|y_4\|\|y_3)$
return $\mathbf{c}$		$\mathbf{r}$	$:= y_2 \oplus Hash(y_1)$
		$\bar{\mathbf{m}}\|\|Const'$	$:= Gen(\mathbf{r}) \oplus y_1$
		If	$Const' = Const$
			return $Prep^{-1}(\bar{\mathbf{m}})$
		Otherwise	reject $\mathbf{c}$

The lengths of y_3, y_4, and y_5 are as follows.

- $Len(y_3) = \lfloor \frac{lm}{8} \rfloor$ bytes.
- $Len(y_4) = \lfloor \left(\frac{r(r-1)}{2} + r(m+n-2r) \right) / 8 \rfloor$ bytes.
- $Len(y_5) = Len(\bar{\mathbf{m}}) + Len(Const) + Len(\mathbf{r}) - Len(y_4) - Len(y_3)$ bytes.
- If $Len(\bar{\mathbf{m}}) + Len(Const) + Len(r) = Len(y_4) + Len(y_3)$, remove y_5.

Referring to [14], it is possible to use the approach in [11] which permits that no information is given in the case of decryption failure. This approach is used in NTRU and the MDPC code based cryptosystem as well.

The function *Conv*

Conv is a function that takes as input a vector $\mathbf{y} \in \mathbb{F}_{q^m}^k$ and outputs a vector $\mathbf{e} \in \mathbb{F}_{q^m}^n$ of rank r. The idea is to use a fixed basis $\beta = \{\beta_1, \beta_2, \ldots, \beta_m\}$ of $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$ to express $\mathbf{y}$ as a matrix with entries in $\mathbb{F}_q$ and use this to form an $m \times n$ matrix E of rank r over $\mathbb{F}_q$. From this matrix we can form the vector $\mathbf{e}$. This approach is similar to Procedure $\mathcal{P}$ given in [2].

The matrix E will be constructed in the following way. Let D be an $r \times r$ matrix of rank r, E_1 be an $(m-r) \times r$ matrix, and E_2 be an $r \times (n-r)$ matrix over $\mathbb{F}_q$. Then the matrix

$$E = \begin{bmatrix} D & DE_2 \\ E_1 & E_1E_2 \end{bmatrix}$$

is of rank r over $\mathbb{F}_q$. Berger and Loidreau [2] pointed out that it is important to ensure randomness in the matrix E. If the random positions are in the matrix D, then the randomness of the error is located on a known subspace of dimension r. This fact can be used for message resend attack. To avoid this, they suggested to use Rijndael S-box, an invertible function which takes in input a byte and return a byte in the output. This S-box has good

diffusion and non-linearity properties. The information bits and the random bits must be spread in bytes such that each byte contains at least one random bit. We apply the Rijndael S-box to each byte and then we put these in D, E_1 and E_2.

Let $\mathbf{y} = (y_1, y_2, \ldots, y_k) \in \mathbb{F}_{q^m}^k$. For the fixed basis β, consider the function $f : \mathbb{F}_{q^m} \to \mathbb{F}_q^m$ such that $f(\sum_{i=1}^m a_i\beta_i) = (a_1, a_2, \ldots, a_m)$. Form the vector $\bar{\mathbf{y}} = (f(y_1), f(y_2), \ldots, f(y_k)) := (\bar{y}_1, \bar{y}_2, \ldots, \bar{y}_{km})$.

Similar to a construction in [12], we construct D by filling the diagonal with 1's and the entries below the diagonal by the first $\frac{r(r-1)}{2}$ entries of $\bar{\mathbf{y}}$ in a lower triangular form as follows:

$$D = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ \bar{y}_1 & 1 & 0 & \cdots & 0 \\ \bar{y}_r & \bar{y}_2 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \bar{y}_{\frac{r(r-1)}{2}} & \cdots & \bar{y}_{2r-3} & \bar{y}_{r-1} & 1 \end{bmatrix}.$$

Next, the matrices E_1 and E_2 are filled with the remaining entries of $\bar{\mathbf{y}}$, i.e.,

$$E_1 = \begin{bmatrix} \bar{y}_{\frac{r(r-1)}{2}+1} & \cdots & \bar{y}_{\frac{r(r-1)}{2}+r} \\ \vdots & \ddots & \vdots \\ \bar{y}_{\frac{r(r-1)}{2}+r(m-r-1)+1} & \cdots & \bar{y}_{\frac{r(r-1)}{2}+r(m-r)} \end{bmatrix}$$

$$E_2 = \begin{bmatrix} \bar{y}_{\frac{r(r-1)}{2}+r(m-r)+1} & \cdots & \bar{y}_{\frac{r(r-1)}{2}+r(m-r)+(n-r)} \\ \vdots & \ddots & \vdots \\ \bar{y}_{\frac{r(r-1)}{2}+r(m-r)+(r-1)(n-r)+1} & \cdots & \bar{y}_{\frac{r(r-1)}{2}+r(m-r)+r(n-r)} \end{bmatrix}$$

where $y_i = 0$ whenever $i > km$. Therefore we have

$$\mathbf{e} = \begin{bmatrix} \beta_1 & \beta_2 & \ldots & \beta_m \end{bmatrix} \cdot E = \left(\sum_{i=1}^m E_{i1}\beta_i, \sum_{i=1}^m E_{i2}\beta_i, \ldots, \sum_{i=1}^m E_{in}\beta_i \right) \in \mathbb{F}_{q^m}^n.$$

Note that the length of $\bar{\mathbf{y}}$ is

$$\lfloor \Big(\frac{r(r-1)}{2} + r(m+n-2r) \Big) / 8 \rfloor \text{ bytes.}$$

The inverse $Conv^{-1}$ is simple, which is the reverse of the previous construction. If the error vector $\mathbf{e} = (e_1, e_2, \ldots, e_n)$ is obtained using the decoding algorithm, we can construct the matrix $E = \left[f(e_1)^T, f(e_2)^T, \ldots, f(e_n)^T \right]$ using the fixed basis of $\mathbb{F}_{q^m}$. This matrix should be of the form $E = \begin{bmatrix} D & DE_2 \\ E_1 & E_1E_2 \end{bmatrix}$. From the nonzero entries in lower triangular matrix D except the diagonal along with the entries of E_1 and E_2, the vector $\bar{\mathbf{y}} = (\bar{y}_1, \bar{y}_2, \ldots, \bar{y}_{km})$ can be obtained. So we recover $\mathbf{y}$ as

$$\mathbf{y} = \left(\sum_{i=1}^m \bar{y}_i\beta_i, \sum_{i=1}^m \bar{y}_{m+i}\beta_i, \ldots, \sum_{i=1}^m \bar{y}_{(k-1)m+i}\beta_i \right).$$

8.2 Practical security

Rank metric code based cryptosystem relies on the rank syndrome decoding (RSD) problems.

Rank Syndrome Decoding (RSD) Problem
Let H be a $(n-k) \times n$ matrix over $\mathbb{F}_{q^m}$ with $k \leq n$, $s \in \mathbb{F}_{q^m}^k$ and r an integer. Find $\mathbf{x}$ such that $Rank(\mathbf{x}) = r$ and $H\mathbf{x}^t = s$.

This problem has been proved to be NP-hard with a randomized reduction [18]. The first proposed McEliece cryptosystem based on rank metric codes was the Gabidulin-Paramonov-Tretjakov (GPT) cryptosystem [13]. But this was broken by Overbeck [26], exploiting the strong algebraic structure of Gabidulin codes. LRPC code based cryptosystem is not the case of GPT cryptosystem since its security relies on the same problem to find a low rank weight codeword in a linear code under no structure.

As the code parameters increase, the complexity of practical attacks also increase very fast since counting the number of possible supports of size r for a rank code of length n over $\mathbb{F}_{q^m}$ is the same as counting the number of subspaces of dimension r in $\mathbb{F}_{q^m}$ [6]. The associated decoding problem, named MinRank, is known to be NP-complete [4].

8.2.1 Direct attacks on the message

There are two types of attacks on the LRPC cryptosystem: direct attacks on the message and structural attacks. Direct attacks on the message try to recover directly the message by finding the error $\mathbf{e}$ of rank r with combinatorial and algebraic attacks. When parameters satisfy the required conditions, combinatorial or algebraic attacks are the most efficient attacks.

• Combinatorial attacks
The performance of the attack in [5] is better when q is small, especially $q = 2$. When q increases and when n and k are not too small, the decoding complexity $(nr+m)^3 q^{(m-r)(r-1)}$ increases rapidly. Ourivski and Johannson's attack shows an improved performance having a complexity of $(k+r)^3 r^3 q^{(r-1)(k+1)}$ [25]. The complexities of these two attacks were generalized to $(n-k)^3 m^3 q^{(r-1)\lfloor \frac{(k+1)m}{n} \rfloor}$ by Gaborit et al. [15], considering n in the complexity when the previous two attacks did not. This new proposed attack deployed the support of a codeword and applied the Information Set Decoding [1] in rank metric sense. The combinatorial attack complexity of McNie is presented in Table 8.2 and calculated for proposed parameters in 8.3 and 8.4 after taking log base q.

• Algebraic attacks
This attack is natural for rank metric case since it is independent of q and in some cases, also independent of m. Thus this attack is most useful when q increases. There are several types of algebraic equations settings to try to solve a multivariate system with Gröbner basis. In 2006 Levy and Perret considered directly the RSD problem, taking the support E of the error as unknowns and the error coordinates regarding E [22]. This algebraic attack used Gröbner basis with complexity lower bound given by $q^{r\lceil \frac{r(k+1)-(n+1)}{r} \rceil}$. However, it is known

that the complexity bounds of these attacks are too huge for practical use. There are also the Kernel attack [8] and the minor attack which uses minors of matrices to get multivariate equations of degree $r+1$ over $\mathbb{F}_q$ [7]. Recently there is an attack using q-polynomials and the annihilator, giving multivariate sparse equations of degree q^{r+1} on the large field $\mathbb{F}_{q^m}$ [15]. The algebraic attack complexity of McNie is presented in Table 8.2 and calculated for proposed parameters in 8.3 and 8.4 after taking *log* base q.

8.2.2 Structural attacks

Structural attacks tries to attack directly the structure of the public key to recover the secret key. The following two attacks use the structure of LRPC codes to obtain the secret key.

• **Known attacks on the LRPC cryptosystem** [16]
The fact that all the elements of the LRPC matrix H belong to the same subspace $\mathcal{F}$ of rank d can be used. For the dual code D generated by H, all the coordinates of $\mathbf{x} \in D$ belong to $\mathcal{F}$. By rewriting $\mathbf{x} = \sum_{i=1}^{n-k} a_i H_i$ for $a_i \in \mathbb{F}_q$ where the $n-k$ rows of H are denoted by $H_i (1 \leq i \leq n-k)$ and determining d a_i's in $\mathbb{F}_q$, it allows $\mathbf{x}$ to have $\lfloor (n-k)/d \rfloor$ coordinates of zeros since H has the weight of d. With a high probability, this vector $\mathbf{x}$ lies in the dual code D and we may assume the first $\lfloor (n-k)/d \rfloor$ coordinates of $\mathbf{x}$ are all zeros without loss of generality. Now the structural attack can be done to LRPC code by choosing the subcode D' of D by puncturing the first $\lfloor (n-k)/d \rfloor$ columns of D. Then D' will be an $[n - \lfloor (n-k)/d \rfloor, n-k-\lfloor (n-k)/d \rfloor]$ code which contains a codeword of rank d.

This attack uses the structure of the LRPC matrix and the attacker only needs to find a subcode which contains at least one codeword of rank d. However the computational cost of this attack is exponential. There is a result with a slightly reduced cost using the cyclicity to decrease the number of columns of the attacked matrix. The attacker can remove columns corresponding to zeros of a small weight vector of the secret key and try to recover it. This attack is equivalent to the attack for NTRU [20] and for MDPC codes based cryptosystem[23].

• **Hauteville and Tillich's attacks** [19]
McEliece cryptosystem based on quasi-cyclic or quasi-monoidic codes can be attacked by reducing the size of the code by adding coordinates which belong to the same orbit of the automorphism group, called the "folding" process [9, 10]. This process is applied to quasi-cyclic, quasi-dyadic, alternant or Goppa codes to attack the cryptosystem for key recovery. It was shown that the same method can be used for quasi-cyclic LRPC codes to obtain a code of much smaller size but have in its dual some low weight codewords. Then the decoding algorithm in [15] can be applied to find low weight codewords in more efficient way than that of the original code. Hauteville and Tillich show their attacks are efficient for double circulant LRPC based system, especially when the polynomial in which the folding process is generalized can be factored. Using this method, one of the proposed parameters in [14] got broken by an attack of complexity $2^{43.6}$. However McNie is secure against this attack since the public key G' is randomly generated so that it can be independent of the secret key H. Thus an attacker trying to recover H from G gains no information using this attack.

8.2.3 Security of McNie

In McNie, since the public code generated by G' is not related to the secret code generated by H, attacking G' does not in any way expose the private code. Thus, finding a low weight codeword in the dual of G' is useless since G' is randomly chosen, and in general, not LRPC. This allows us to have freedom on choosing our low rank d as it does not affect the public keys. Since LRPC decoding can be used as long as $rd \leq n-k$, we can choose d small enough and r high enough for increased security.

An attacker may, on the other hand, attack the public matrix F to obtain H. Since G' is known, an attacker can proceed with decomposing F to $F = G'H_0$. This would yield $q^{m(n-l)(n-k)}$ possible solutions H_0. However, because of scrambler matrix S, a solution $H_0 = H^T S$ may not be a low rank so LRPC decoding may not be successful. The number of possible solutions is calculated and shown in 10th column of Tables 8.3 and 8.4 after taking log base q.

For the message attack, in general, if G' has a smaller error-correction capability compared to H, then decoding using G will fail. Also, since G' is randomly generated, this is the rank-RSD problem. In this proposal, we can select good parameters in order to increase the decoding failure probability of using the public matrix G' while keeping a low decoding failure probability using the private matrix H. An attacker may also use $\mathbf{c_2}$ to recover $\mathbf{m}$ by solving linear systems. However McNie is designed to have $q^{m(l-(n-k))}$ possible solutions. Although $\mathbf{c_2} = \mathbf{m}F$ resembles the ciphertext from the Niederreiter cryptosystem, notice that there is no restriction on the weight of the message $\mathbf{m}$ so attacking the Niederreiter cryptosystem does not threaten the security of McNie.

Table 8.2: Complexity of known attacks to McNie

Combinatorial Attack	Algebraic Attack	McNie Attacks
$(n-l)^3 m^3 q^{(r-1)\lfloor \frac{(l+1)m}{n} \rfloor}$	$q^{r\lceil \frac{r(l+1)-(n+1)}{r} \rceil}$	$q^{m(l-(n-k))}$

Table 8.3: Complexity of known attacks to suggested parameters using 3-quasi-cyclic LRPC codes

n	k	l	d	r	m	q	Combi. Attack	Algeb. Attack	McNie Attack	Key Size (bits)	Security
84	56	56	3	4	29	2	86	124	812	2436	80
96	64	64	3	4	29	2	86	164	928	2784	80
93	62	62	3	5	37	2	130	225	1147	3441	128
105	70	70	3	5	37	2	131	250	1295	3885	128
111	74	74	3	7	41	2	194	413	1517	4551	192
123	82	82	3	7	41	2	194	462	1681	5043	192
111	74	74	3	7	59	2	267	413	2183	6549	256
141	94	94	3	9	47	2	281	720	2209	6627	256

Table 8.4: Complexity of known attacks to suggested parameters using 4-quasi-cyclic LRPC codes

n	k	l	d	r	m	q	Combi. Attack	Algeb. Attack	McNie Attack	Key Size (bits)	Security
48	24	36	2	4	29	2	91	100	348	1740	80
56	28	42	2	4	37	2	111	116	518	2590	80
60	30	45	3	5	37	2	139	170	555	2775	128
72	36	54	3	5	37	2	140	205	666	3330	128
76	38	57	3	7	41	2	214	315	738	3895	192
84	42	63	3	7	41	2	215	364	861	4305	192
76	38	57	3	7	53	2	269	329	1007	5035	256
88	44	66	3	8	47	2	275	448	1034	5170	256

Table 8.5: Key-size (bits) comparison with other code-based cryptosystems

Security Level	McNie		DC-LRPC [17]	DC-MDPC [23]	QD-Goppa [24]	Goppa [3]
	3-quasi	4-quasi				
80	**2436**	**1740**	1681	4801	20480	460647
128	**3441**	**2775**	2809	9857	32768	1537536
192	**4551**	**3895**	-	-	45056	4185415
256	**6549**	**5035**	-	32771	65536	7667855

Table 8.6: Comparison of key sizes (bits)

Security Level	McNie		NTRU	RSA	ECC	ECC AWC
	3-quasi	4-quasi				
80	**2436**	**1740**	3487	1024	163	112210
128	**3441**	**2775**	4939	3072	256	277280
192	**4551**	**3895**	6523	7680	384	936618
256	**6549**	**5035**	8173	15360	512	1595434

Chapter 9

Advantages and Limitations (2.B.6)

- Advantages

  1. A random code is used in the encryption so that McNie is secure against structural and information set decoding attacks.
  2. McNie provides significant improvement in security compared to the GRSTZ cryptosystem with the same parameters. Table 9.1 shows the complexities in bits of known attacks on both cryptosystems that uses an $(n-k) \times n$ parity check matrix of an LRPC code of weight d in the private key. Here we use the suggested parameters for McNie using 4-quasi-cyclic LRPC codes achieving 192-bit and 256-bit security levels.

Table 9.1: McNie vs the GRSTZ cryptosystem of the same parameters

Decryption						Combi. Attack		Algeb. Attack	
n	k	d	r	m	q	**McNie**	QC-LRPC	**McNie**	QC-LRPC
76	38	3	7	41	2	**214**	158	**315**	196
76	38	3	7	53	2	**269**	194	**329**	196

  3. The GRSTZ cryptosystem can be shown to be a special case of McNie. Suppose H is the parity check-matrix for a quasi-cyclic LRPC code, as described in Section 2.5 and G be its generator matrix. If $G' = RG$ for some invertible matrix R and P is the identity matrix, then $F = RGH^TS = 0$. Hence $\mathbf{c_2} = 0$ and $\mathbf{c_1} = \mathbf{m}RG + \mathbf{e}$ is the same as the ciphertext for the GRSTZ cryptosystem. McNie uses the same decryption algorithm as the GRSTZ cryptosystem. Thus we conclude that McNie is harder to attack than the GRSTZ cryptosystem.
  4. Comparing with other encryption schemes, McNie provides significantly smaller public key sizes that increase at a more gradual rate as security level increases.
  5. McNie can use various kinds of known block codes as inputs even though McEliece cryptosystem based on those codes were broken. The reason is that McNie uses a random code seems more secure than McEliece against structural and ISD attacks. Therefore, studying McNie will open more research areas in the near future.

- Limitations

  LRPC decoding is probabilistic and there is a nonzero probability of failure in decryption that the $n - k$ syndromes does not generate the space $P = \langle E.F \rangle$. It was originally mentioned in the GRSTZ cryptosystem and eventually the GRSTZ cryptosystem also has the second failure problem of $dim(\cap S_i) \neq r$. It was considered as a small probability in GRSTZ, but it was large for some parameters. While this can be easily minimized by adjusting parameters, it comes at the cost of the key size. Our suggested parameters are optimized for key size and low failure probabilities that we think best fit.

  In the IND-CCA2 conversion, encrypting the same message results in a different ciphertext. So in the event of a decryption failure, receiver may ask to resend the message and this does not pose a threat in the security of McNie.

  McNie will be useful for applications such that third party can resend ciphertext several times without knowing the original message and reducing the failure probability in decryption. For example, for `encrypt/3Q_128_1`, the failure probability is 2^{-17} as shown in Table 3.1, which can be further reduced to 2^{-34} by resending the message twice.

Bibliography

[1] Becker, A., Joux, A., May, A. and Meurer, A.: Decoding random binary linear codes in $2n/20$: How $1 + 1 = 0$ improves information set decoding. Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp. 520-536. Springer, Berlin, Heidelberg (2012)

[2] Berger, T. and Loidreau, P.: Designing an efficient and secure public-key cryptosystem based on reducible rank codes. In: Canteaut A., Viswanathan K. (eds) Progress in Cryptology - INDOCRYPT 2004. INDOCRYPT 2004. Lecture Notes in Computer Science, vol 3348, pp. 218-229 Springer, Berlin, Heidelberg (2004)

[3] Bernstein, D.J., Lange, T., and Peters, C.: Attacking and defending the McEliece cryptosystem. In Proceedings of the 2nd International Workshop on Post-Quantum Cryptography, PQCrypto '08, pp. 31–46, Springer-Verlag, Berlin, Heidelberg (2008)

[4] Buss, J.F., Frandsen, G.S. and Shallit, J.O.: The computational complexity of some problems of linear algebra. Journal of Computer and System Sciences 58(3), pp. 572-596 (1999)

[5] Chabaud, F. and Stern, J.: The cryptographic security of the syndrome decoding problem for rank distance codes. ASIACRYPT 1996. LNCS vol. 1163, pp. 368-381. Springer, Heidelberg (1999)

[6] Delsarte, P.H.: Bilinear Forms over a finite field, with applications to coding theory. Journal of Combinatorial Theory Series A 25, pp. 226-241 (1978)

[7] Faugère, J.C., El Din, M.S. and Spaenlehauer, P.J.: Computing loci of rank defects of linear matrices using Gröbner bases and applications to cryptology. Proceedings of the 2010 International Symposium on Symbolic and Algebraic Computation, pp. 257-264. ACM (2010)

[8] Faugère, J.C., Levy-Dit-Vehel, F. and Perret, L.: Cryptanalysis of MinRank. CRYPTO 2008. LNCS vol. 5157, pp. 280-296. Springer Heidelberg (2008)

[9] Faugère, J.C., Otmani, A., Perret, L., de Portzamparc, F. and Tillich, J.P.: Structural weakness of compact variants of the McEliece cryptosystem. IEEE International Symposium on Information Theory - ISIT 2014, pp.1717-1721 (2014)

[10] Faugère, J.C., Otmani, A., Perret, L., de Portzamparc, F. and Tillich, J.P.: Structural cryptanalysis of McEliece schemes with compact keys. Designs, Codes and Cryptography 79(1), pp. 87-112 (2016)

[11] Fujisaki, E. and Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. CRYPTO'99. LNCS vol. 1666, pp. 537-554 (1999)

[12] Gabidulin, E. M., Ourivski, A. V., Honary, B., and Ammar, B.: Reducible rank codes and their applications to cryptography. IEEE Transactions on Information Theory, 49(12), pp. 3289-3293 (2003)

[13] Gabidulin, E.M., Paramonov, A.V. and Tretjakov, O.V.: Ideals over a non-commutative ring and their application in cryptology. Workshop on the Theory and Application of of Cryptographic Techniques, pp. 482-489. Springer, Berlin, Heidelberg (1991)

[14] Gaborit, P., Murat, G., Ruatta, O. and Zémor, G.: Low rank parity check codes and their application to cryptography. The Proceedings of Workshop on Coding and Cryptography (WCC) 2013, Borgen, Norway, pp. 168-180 (2013)

[15] Gaborit, P., Ruatta, O. and Schrek, J.: On the complexity of the rank syndrome decoding problem. IEEE Transactions on Information Theory 62(2), pp. 1006-1019 (2016)

[16] Gaborit, P., Ruatta, O., Schrek, J. and Zémor, G.: New results for rank-based cryptography. AFRICACRYPT 2014: Progress in Cryptology. LNCS vol 8469, pp. 1-12. Springer, Cham (2016)

[17] Gaborit, P., Ruatta, O., Schrek, J., Tillich, J. P., Zémor, G.: Rank based Cryptography: a credible post-quantum alternative to classical crypto. In NIST 2015: Workshop on Cybersecurity in a Post-Quantum World 2015 (2015)

[18] Gaborit, P. and Zémor, G.: On the hardness of the decoding and the minimum distance problems for rank codes. IEEE Transactions on Information Theory 62(12), pp. 7245-7252 (2016)

[19] Hauteville, A. and Tillich, J.P.: New algorithms for decoding in the rank metric and an attack on the LRPC cryptosystem. IEEE International Symposium on Information Theory - ISIT 2015, pp. 2747-2751 (2015)

[20] Hoffstein, J., Pipher, J. and Silverman, J.H.: NTRU: A ring-based public key cryptosystem. International Algorithmic Number Theory Symposium, pp. 267-288. Springer, Berlin, Heidelberg (1998)

[21] Kobara, K. and Imai, H.: Semantically secure McEliece public-key cryptosystems-conversions for McEliece PKC. Public Key Cryptography vol. 1992, pp. 19-35 (2001)

[22] Levy-dit-Vehel, F. and Perret, L.: Algebraic decoding of rank metric codes. Proceedings of YACC06. (2006)

[23] Misoczki, R., Tillich, J. P., Sendrier, N. and Barreto, P. S.: MDPC-McEliece: New McEliece variants from moderate density parity-check codes. IEEE International Symposium on Information Theory - ISIT 2013, pp. 2069-2073 (2013)

[24] Misoczki, R., and Barreto, P. S.: Compact McEliece keys from Goppa codes. In Selected Areas in Cryptography, pp. 376–392 (2009)

[25] Ourivski, A.V., and Johansson, T.: New technique for decoding codes in the rank metric and its cryptography applications. Problems of Information Transmission 38(3), pp. 237-246 (2002)

[26] Overbeck, R.: A new structural attack for GPT and variants. Mycrypt 2005: Progress in Cryptology. LNCS vol. 3715, pp. 50-63 (2005)