Analysis of Layered ROLLO-I: A BII-LRPC code-based KEM

Seongtaek Chee[1], Kyung Chul Jeong[1], Tanja Lange[2], Nari Lee[1], Alex Pellegrini[2], and Hansol Ryu[1]

[1] The Affiliated Institute of ETRI, Daejeon 34044, Republic of Korea
{chee, jeongkc, narilee, hansolryu}@nsr.re.kr
[2] Eindhoven University of Technology, 5612 AZ Eindhoven, Netherlands
tanja@hyperelliptic.org, alex.pellegrini@live.com

Abstract. We analyze Layered ROLLO-I, a code-based cryptosystem published in IEEE and submitted to the Korean post-quantum cryptography competition. Four versions of Layered ROLLO-I have been proposed in the competition. We show that the first two versions do not provide the claimed security against rank decoding attacks and give reductions to small instances of the original ROLLO-I scheme, which was a candidate in the NIST competition and eliminated there due to rank decoding attacks. Finally, we provide two efficient message recovery attacks, affecting every security level of the first three versions of Layered ROLLO-I and security levels 128 and 192 of the fourth version.

Keywords: Post-quantum cryptography · code-based cryptography · rank-metric code · BII-LRPC code · Layered ROLLO-I.

1 Introduction

Rank metric codes were introduced to cryptography by Gabidulin, Paramonov, and Tretjakov at Eurocrypt'91 [5] but attacked and broken 10 years later by Overbeck in several papers, covered in [12]. The NIST competition on post-quantum cryptography saw a revival of rank-metric codes in rounds 1 and 2 until some new algebraic attacks were found near the end of round 2. These attacks did not completely break the systems and larger parameters were proposed that would resist the new attacks, but the attacks showed that rank-metric codes were not mature enough to be used. Furthermore, the larger parameters would have hurt the performance of the systems. Consequently, NIST deselected all rank-metric-based designs from advancing to round 3.

In this paper, we analyze a blockwise interleaved ideal low-rank parity-check (BII-LRPC) code-based KEM, which was proposed by Kim, Kim, and No in [10] and submitted to the KpqC Competition under the name Layered ROLLO-I [6]. Layered ROLLO-I is a modified version of the NIST candidate ROLLO [1], and particularly of ROLLO-I. Layered ROLLO-I adds additional structure to increase the length of the codewords that an attacker is faced with while at the same time permitting the legitimate receiver, using the secret key, to peel off this layer of structure and then to perform rank decoding with parameters which are even smaller than in ROLLO-I, thus increasing performance.

In this paper we describe attacks on four versions of Layered ROLLO-I that have been released subsequently to the communication of our analyses on the KpqC bulletin. We show how to reduce every instance of the first two versions to an instance of original ROLLO-I at the smaller parameter size that Layered ROLLO-I uses internally. This shows that the additional structure does not add any security. As a consequence, the parameter sets proposed for Layered ROLLO-I offer less security than the parameter sets for the corresponding levels in the original ROLLO-I. After these two reduction attacks we show a message recovery attack that works very efficiently for all parameter sets in the first three versions and for two out of three levels for the fourth version.

This paper is organized as follows. The next section is dedicated to the needed notation and background. For the sake of simplicity, we will refrain from introducing the entire framework of rank metric codes, (ideal and blockwise interleaved ideal) low-rank parity check (LRPC) codes, since these notions will not be directly used in the attacks and analysis. In Section 3.1, we give the specification of Layered ROLLO-I and propose an attack reducing Layered ROLLO-I to ROLLO-I and recalculate costs of RSD attacks following the improvements in [3,4,2]. In Section 3.2, we describe the first Modified Layered ROLLO-I (MLR1) system. We also show that we can adapt the attack in Section 3.1 to reduce this system to ROLLO-I and further improve on RSD attacks complexities. Section 4 introduces two message recovery attacks. In Section 4.1, we describe the second modified Layered ROLLO-I (MLR2) that resists the reduction attacks presented in Section 3. We propose an efficient message recovery attack that can be applied to all the versions of Section 3 and MLR2. Finally, in Section 4.3 we introduce the third modified Layered ROLLO-I (MLR3) and a message recovery attack. This attack recovers messages efficiently for security levels 128 and 192 of MLR3.

2 Notation and background

In the specifications of this paper, we will follow the notation of [10] with minor changes. For an element $\mathbf{x} \in \mathbb{F}_{q^m}^n$ let $\mathsf{wt}_R(\mathbf{x})$ denote the rank weight of $\mathbf{x}$, which is defined as the rank of the $m \times n$-matrix over $\mathbb{F}_q$ containing as columns the representations of the entries of $\mathbf{x}$ relative to some fixed basis of $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$.

Denote by $S_w^n(\mathbb{F}_{q^m})$ the set of vectors of length n and rank weight w over $\mathbb{F}_{q^m}$:

$$S_w^n(\mathbb{F}_{q^m}) = \{\mathbf{x} \in \mathbb{F}_{q^m}^n \mid \mathsf{wt}_R(\mathbf{x}) = w\}.$$

Let $\mathbf{w} = \mathbf{c} + \mathbf{e}$ for some codeword $\mathbf{c}$ and error $\mathbf{e}$ with $\mathsf{wt}_R(\mathbf{e}) \le r$ and let $\mathbf{s} = \mathbf{w}H^T$ be the syndrome of $\mathbf{w}$ using a parity-check matrix H of the code.

The Rank Support Recovery ($\mathsf{RSR}(F, \mathbf{s}, r)$) algorithm is used as a decoder in the decapsulation procedures of ROLLO-I and the follow-up designs. It recovers the support E of (the $\mathbb{F}_q$-linear subspace of $\mathbb{F}_{q^m}$ generated by) the error vector $\mathbf{e}$ given the support F of the dual code[3], the syndrome $\mathbf{s}$, and the rank r of the

[3] Thus F defines the parity check for the code.

error. This corresponds to actually finding the error coordinates, by solving a linear system of equations (see p. 13 of the ROLLO specification [1]).

Let $P(x) \in \mathbb{F}_{q^m}[x]$ be a polynomial of degree n. We can identify the vector space $\mathbb{F}_{q^m}^n$ with the ring $\mathbb{F}_{q^m}[x]/(P(x))$, where $(P(x))$ is the ideal of $\mathbb{F}_{q^m}[x]$ generated by $P(x)$. Given $\mathbf{u} = (u_0, \dots, u_{n-1}) \in \mathbb{F}_{q^m}^n$, denote by $\mathbf{u}(x) \in \mathbb{F}_{q^m}[x]$ the polynomial $\mathbf{u}(x) = \sum_{i=0}^{n-1} u_i x^i$. Given $\mathbf{u}, \mathbf{v} \in \mathbb{F}_{q^m}^n$, we define their product $\mathbf{uv}$ as the unique vector $\mathbf{w} \in \mathbb{F}_{q^m}^n$ such that $\mathbf{w}(x) = \mathbf{u}(x)\mathbf{v}(x) \bmod P(x)$. Similarly, we define $Q\mathbf{u} = Q(x)\mathbf{u}(x) \bmod P(x)$ for $Q(x) \in \mathbb{F}_{q^m}[x]$ and $\mathbf{u}^{-1}$ for $\mathbf{u}(x)$ invertible modulo $P(x)$.

2.1 ROLLO-I [1]

We give a simple description of ROLLO-I, which is the base of Layered ROLLO-I.

The values (q, n, m, r, d, P) are the system parameters, where q, n, m, r, d are integers and $P(x) \in \mathbb{F}_{q^m}[x]$ is a primitive polynomial of degree n.

- KeyGen:
  - Pick random $\mathbf{x}, \mathbf{y} \in S_d^n(\mathbb{F}_{q^m})$.
  - Set $\mathbf{h}(x) = \mathbf{x}(x)^{-1}\mathbf{y}(x) \bmod P(x)$.
  - Return $\mathsf{pk} = \mathbf{h}$ and $\mathsf{sk} = (\mathbf{x}, \mathbf{y})$.
- Encap(pk):
  - Pick random $(\mathbf{e}_1, \mathbf{e}_2) \in S_r^{2n}(\mathbb{F}_{q^m})$.
  - Set $E = \langle \mathbf{e}_1, \mathbf{e}_2 \rangle$, where $\langle \mathbf{e}_1, \mathbf{e}_2 \rangle$ denotes the $\mathbb{F}_q$-vector space spanned by the columns of $\mathbf{e}_1$ and $\mathbf{e}_2$ (interpreted as vectors in $\mathbb{F}_q^m$).
  - Return $K = \mathsf{hash}(E)$ and $\mathbf{c}(x) = \mathbf{e}_1(x) + \mathbf{e}_2(x)\mathbf{h}(x) \bmod P(x)$.
- Decap(sk):
  - Set $\mathbf{s}(x) = \mathbf{x}(x)\mathbf{c}(x) \bmod P(x)$, $F = \langle \mathbf{x}, \mathbf{y} \rangle$ and $E = \mathsf{RSR}(F, \mathbf{s}, r)$.
  - Return $K = \mathsf{hash}(E)$.

3 Reduction Attacks

Layered ROLLO-I uses a structure of layer. Due to the special structure, the performance has improved by 30-70% compared to ROLLO-I. In this section, we give a simple description of Layered ROLLO-I and its variant and show that the layer can be removed by exploiting public information. As a result, the security of each algorithm is reduced to that of ROLLO-I for the small parameters inside the layer, which gives far lower complexity than was suggested in [10].

3.1 Layered ROLLO-I

The values (q, n, m, r, d, b, P) are the system parameters, where q, n, m, r, d, b are integers, with n a multiple of b, and $P(x) \in \mathbb{F}_{q^m}[x]$ is a primitive polynomial of degree n/b. For all parameter sets defined in [10] we have $b = 2$ and in any case $b < n/b$. The map $\Psi : \mathbb{F}_{q^m}[x]/(P(x)) \to \mathbb{F}_{q^m}[x]/(P(x)^b)$ casts polynomials of the first quotient into the second quotient by mapping the input to the unique polynomial of degree $< n/b$ that is congruent to it modulo $P(x)^b$. Similarly, the map $\Omega : \mathbb{F}_{q^m}[x]/(P(x)^b) \to \mathbb{F}_{q^m}[x]/(P(x))$ reduces the input modulo $P(x)$. Since $P(x)^b$ is a multiple of $P(x)$ these maps are well-defined.

- KeyGen:
  - Pick random $\mathbf{x}, \mathbf{y} \in S_d^{n/b}(\mathbb{F}_{q^m})$.
  - Pick random invertible $P_I(x) \in \mathbb{F}_{q^m}[x]/(P(x))$ of degree $(b-1)$.
  - Pick random $P_O(x), P_N(x) \in \mathbb{F}_{q^m}[x]/(P(x)^b)$ of degree n, with $P_O(x)$ invertible (this last restriction is not stated but is required for functionality).
  - Set $\mathbf{z}(x) = P_I(x)\mathbf{x}(x)^{-1}\mathbf{y}(x) \bmod P(x)$.
  - Set $P_P(x) = P_O(x)\Psi(P_I(x)) \bmod P(x)^b$ and $P_H(x) = P_O(x)\Psi(\mathbf{z}(x)) + P_N(x)P(x) \bmod P(x)^b$.
  - Return $\mathsf{pk} = (P_P, P_H)$ and $\mathsf{sk} = (\mathbf{x}, \mathbf{y}, P_O, P_I)$.
- Encap(pk):
  - Pick random $E = \langle \mathbf{e}_1, \mathbf{e}_2 \rangle$ with $\mathbf{e}_1, \mathbf{e}_2 \in S_r^{n/b}(\mathbb{F}_{q^m})$, each corresponding to a polynomial of degree $< n/b - b$.
  - Set $P_{E_1}(x) = \Psi(\mathbf{e}_1(x))$ and $P_{E_2}(x) = \Psi(\mathbf{e}_2(x))$.
  - Compute $\mathbf{c}(x) = P_P(x)P_{E_1}(x) + P_H(x)P_{E_2}(x) \bmod P(x)^b$.
  - Return $K = \mathsf{hash}(E)$ and $\mathbf{c}$.
- Decap(sk):
  - Compute $P_C(x) = P_O(x)^{-1}\mathbf{c}(x) \bmod P(x)^b$.
  - Compute $\mathbf{c}'(x) = P_I(x)^{-1}\Omega(P_C(x)) \bmod P(x)$.
  - Decode $E = \mathsf{RSR}(\langle \mathbf{x}, \mathbf{y} \rangle, \mathbf{x}\mathbf{c}', r)$.
  - Return $K = \mathsf{hash}(E)$.

The attacker thus faces polynomials modulo P^b as public key and as ciphertexts. These correspond to vectors of length n. The decapsulation process removes the outer layer so that the RSR step works modulo P and thus on vectors of length n.

Reduction of Layered ROLLO-I to ROLLO-I We propose a new reduction of Layered ROLLO-I to ROLLO-I by using exclusively the public key of the former. To start with, notice that P_O must have an inverse modulo P^b. This has not been declared in the specification but the decapsulation process requires P_O^{-1}. If not, decapsulation fails. Also, P_I is irreducible of degree $(b-1) < n/b = \deg P$, so it has an inverse modulo P and thus $\Psi(P_I)$ is invertible modulo P^b. Therefore, we can invert P_P modulo P^b and compute $P_P(x)^{-1}P_H(x)$ as

$$\Psi(P_I(x))^{-1}\Psi(\mathbf{z}(x)) + P_P(x)^{-1}P_N(x)P(x) + k(x)P(x)^b \qquad (1)$$

for some $k(x) \in \mathbb{F}_{q^m}[x]$. Since P divides P^b, we can reduce the equation modulo P, obtaining

$$\begin{aligned} P_P(x)^{-1}P_H(x) &\equiv \Psi(P_I(x))^{-1}\Psi(\mathbf{z}(x)) \\ &\equiv \Psi(P_I(x))^{-1}P_I(x)\mathbf{x}(x)^{-1}\mathbf{y}(x) \\ &\equiv \mathbf{x}(x)^{-1}\mathbf{y}(x) \quad \bmod P(x), \end{aligned}$$

where the second equivalence follows from that $\Psi(\mathbf{z}(x)) \equiv P_I(x)\mathbf{x}(x)^{-1}\mathbf{y}(x) \bmod P(x)$, and the last equivalence from $\Psi(P_I(x))^{-1} \equiv P_I(x)^{-1} \bmod P(x)$.

This shows that the public key of (q, n, m, r, d, b)-Layered ROLLO-I can be reduced to the public key of $(q, n/b, m, r, d)$-ROLLO-I. The same can be done for ciphertexts by computing $P_P(x)^{-1}\mathbf{c}(x) = P_P(x)^{-1}(P_P(x)P_{E_1}(x) + P_H(x)P_{E_2}(x)) = P_{E_1}(x) + \mathbf{y}(x)\mathbf{x}(x)^{-1}P_{E_2}(x) \bmod P(x)^b$, which is exactly a ROLLO-I ciphertext.

Estimates for the security of Layered ROLLO-I: Layered ROLLO-I suggests an attack that removes the layer of a BII-LRPC code using exhaustive search and applies a structural attack to an instance of $(q, n/b, m, r, d)$-ROLLO-I [10]. The suggested cost of the attack is shown in the third column of Table 1. However, the calculation is wrong and furthermore, the formula given in [10] has a typo. The correct formula for the attack is given below.

$$S'_S = \left(\frac{n}{b}\right)^3 m^3 q^{(b-1)m + d\lceil \frac{m}{2} \rceil - m - \frac{n}{b}}. \tag{2}$$

While the correct formula increases attack complexity when compared to the suggested one in [10], the accurate computation yields significantly lower complexity, which are 65, 112, and 131 bits, respectively.

Now we consider the attacks in [3,4,2], where we discard the options in [4] that have been proved too optimistic in [3]. Since Layered ROLLO-I did not consider applying these three attacks on the original parameters directly, we recompute the costs of rank decoding attacks, finding out that the proposed parameters are not suitable for the requested security levels. The most efficient values of these attack costs are reported in the fourth column of Table 1. The last column reports the cost of these attacks on the system after our reduction.

Security	(q, n, m, r, d, b)	Cost [10]	Cost	Cost red.
128	$(2, 148, 67, 3, 2, 2)$	130.83	48.76 [4]	40.65 [4]
192	$(2, 172, 79, 4, 3, 2)$	199.19	66.21 [4]	55.16 [4]
256	$(2, 212, 97, 5, 3, 2)$	274.98	85.68 [4]	72.05 [4]

Table 1. Suggested parameters and values of the $\log_2$ of attack costs for Layered ROLLO-I's suggested parameters. Cost [10] refers to the cost stated in the paper introducing the system; references after the costs refer to the publication which has the best attack on these parameters.

3.2 First Modified Layered ROLLO-I (MLR1)

This subsection extracts the description of the modified system MLR1 from [7]. The designers modified the system to overcome the reduction in Section 3.1 by replacing the two moduli P and P^b by two primitive polynomials P_1 and P_2 of degree n_1 and n_2, respectively. Because they are primitive they are in particular irreducible and thus coprime. In this setting, one cannot simply reduce equation (1) modulo P_1 as the term $k(x)P_2(x)$ would not vanish which seems to stop the attack.

However, this might make it seem like decapsulation cannot recover $(\mathbf{e}_1, \mathbf{e}_2)$ either because the moduli are incompatible. The KEM works around this problem by reducing the degrees of $\mathbf{e}_1$ and $\mathbf{e}_2$. In this setting, Ω first lifts to $\mathbb{F}_{q^m}[x]$ choosing the unique polynomial of degree less than n_2 and then reduces modulo P_1, Ψ similarly lifts to $\mathbb{F}_{q^m}[x]$ choosing the unique polynomial of degree less than n_1 and then considers this polynomial modulo P_2. Given that $n_2 > n_1$ no reduction is needed.

The values $(q, n_1, n_2, d_I, m, r, d)$, where $d_I < n_1 < n_2$ are the system parameters. The two polynomials P_1 and P_2 are primitive of degrees n_1 and n_2 respectively. These are not stated among the system parameters but are needed for the functioning of the system. In the following, we assume that P_1 and P_2 are part of the system parameters.

- KeyGen:
  - Pick random $\mathbf{x}, \mathbf{y} \in S_d^{n_1}(\mathbb{F}_{q^m})$.
  - Pick random invertible $P_I(x) \in \mathbb{F}_{q^m}[x]/(P_1(x))$ of degree d_I.
  - Pick random $P_O(x) \in \mathbb{F}_{q^m}[x]/(P_2(x))$.
  - Set $\mathbf{z}(x) = P_I(x)\mathbf{x}(x)^{-1}\mathbf{y}(x) \bmod P_1(x)$.
  - Set $P_P(x) = P_O(x)\Psi(P_I(x)) \bmod P_2(x)$ and $P_H(x) = P_O(x)\Psi(\mathbf{z}(x)) \bmod P_2(x)$.
  - Return $\mathsf{pk} = (P_P, P_H)$ and $\mathsf{sk} = (\mathbf{x}, \mathbf{y}, P_O, P_I)$.

- Encap(pk):
  - Pick random $E = \langle \mathbf{e}_1, \mathbf{e}_2 \rangle$ with $\mathbf{e}_1, \mathbf{e}_2 \in S_r^{n_2}(\mathbb{F}_{q^m})$ each corresponding to a polynomial of degree $< n_2 - n_1 - d_I$.
  - Set $P_{E_1} = \mathbf{e}_1(x)$ and $P_{E_2} = \mathbf{e}_2(x)$. item Compute $\mathbf{c}(x) = P_P(x)P_{E_1}(x) + P_H(x)P_{E_2}(x) \bmod P_2(x)$.
  - Return $K = \mathsf{hash}(E)$ and $\mathbf{c}$.
- Decap(sk):
  - Compute $\mathbf{c}''(x) = P_O(x)^{-1}\mathbf{c}(x) \bmod P_2(x)$.
  - Compute $\mathbf{c}'(x) = P_I(x)^{-1}\Omega(\mathbf{c}''(x)) \bmod P_1(x)$.
  - Decode $E = \mathsf{RSR}(\langle \mathbf{x}, \mathbf{y} \rangle, \mathbf{x}\mathbf{c}', r)$.
  - Return $K = \mathsf{hash}(E)$.

Decapsulation works because

$$\begin{aligned}\mathbf{c}''(x) &= P_O(x)^{-1}(P_P(x)P_{E_1}(x) + P_H(x)P_{E_2}(x)) \\ &= P_O(x)^{-1}(P_O(x)\Psi(P_I(x))P_{E_1}(x) + P_O(x)\Psi(\mathbf{z}(x))P_{E_2}(x)) \\ &= \Psi(P_I(x))P_{E_1}(x) + \Psi(\mathbf{z}(x)P_{E2}(x) \bmod P_2(x)\end{aligned}$$

and the degree of $\Psi(P_I(x))P_{E_1}(x) + \Psi(\mathbf{z}(x))P_{E_2}(x)$ is $< n_2$ by the choice of the error vectors. Hence, $\mathbf{c}''(x) = \Psi(P_I(x))P_{E_1}(x) + \Psi(\mathbf{z}(x)P_{E2}(x)$ in $\mathbb{F}_{q^m}[x]$ i.e., without reduction, and thus the reduction modulo $P_1(x)$ preserves the factors $P_I(x)$ which can then be divided out.

Reduction of $\mathsf{MLR1}$ **to ROLLO-I** Now we will describe a reduction of $\mathsf{MLR1}$. Along the way we compute P_I and P_O, meaning that the system leaks private information.

The idea of the reduction remains the same, observing that $P_H(x)/P_P(x)$ cancels the P_O. However, because of the coprimality of the moduli, we cannot proceed directly from there to reducing modulo P_1. Nevertheless, we know that the polynomials involved have very low degrees. Let $R(x) = P_H(x)/P_P(x) \bmod P_2(x)$ then $\deg(R) < n_2$ and $R(x) = \Psi(\mathbf{z}(x))/\Psi(P_I(x)) \bmod P_2(x)$ with $\deg(\mathbf{z}) < n_1$ and d_I small. Note that the division might cancel common factors of P_I and $\mathbf{z}$, however, given the degrees this is unlikely.

Let M_R be the $(d_I + 1) \times n_2$ matrix over $\mathbb{F}_{q^m}$ representing multiplication of a polynomial of degree up to d_I by R modulo P_2, i.e.

$$M_R = \begin{pmatrix} R(x) \bmod P_2(x) \\ R(x)x \bmod P_2(x) \\ \vdots \\ R(x)x^{d_I} \bmod P_2(x) \end{pmatrix}, \tag{3}$$

where each row consists of the coefficient vector of $R(x)x^i \bmod P_2(x)$ for $i = 0, \dots, d_I$.

Remark 1. Let A be any $n \times m$ matrix, with $n, m \in \mathbf{N}$. We denote by $A[a : b, c : d]$, with $a < b \in [1, n]$ and $c < d \in [1, m]$, the submatrix of A consisting of the rows in the range $[a, b]$ and columns in the range $[c, d]$. We omit a and b, i.e. use $A[:, c : d]$ to denote the submatrix consisting of all the rows and columns in $[c, d]$. Similarly, for all the columns. With this notation $A = A[:, :]$. If $S_1 \subset [1, n]$ and $S_2 \subset [1, m]$ we denote by $A[S_1, S_2]$ the submatrix of A consisting of rows indexed by S_1 and columns indexed by S_2.

The polynomial P_2 is irreducible and thus $\mathbb{F}_{q^m}[x]/(P_2(x))$ defines a field. Note that multiplication by R defines an automorphism of this field, thus the associated matrix M has rank n_2. Therefore, since $M_R = M[1 : d_I + 1, :]$, it has rank $d_I + 1$. Let $\pi : \mathbb{F}_{q^m}^{n_2} \to \mathbb{F}_{q^m}^{d_I+1}$ be the projection of an element of $\mathbb{F}_{q^m}^{n_2}$ onto its first $d_I + 1$ coordinates. Consider

$$\pi(\Psi(P_I(x)))M_R = \Psi(\mathbf{z}(x)) \tag{4}$$

as a linear system of equations in the coefficients of $\Psi(P_I)$ and $\Psi(\mathbf{z})$, where in this case we view $\Psi(\mathbf{z})$ as an element of $\mathbb{F}_{q^m}^{n_2}$ consisting of the unknown coefficients of $\Psi(\mathbf{z})$ and $n_2 - n_1$ trailing zeroes. Note that π does not induce any loss of information due to the degree of $\Psi(P_I)$. Since $\deg(\Psi(P_I)) + n_1 = d_I + n_1 < n_2$, the system has a solution corresponding to the representatives of P_I and $\mathbf{z}$ modulo P_1 (here we remove the Ψ notation as the solutions will have degree lower than n_1).

We can actually compute P_I from a subset of the equations defined by (4). Indeed, $\pi(\Psi(P_I))$ lies in the left kernel of the submatrix of M_R that consists of the last $n_2 - n_1$ columns, meaning that such submatrix has rank at most

d_I, and typically exactly d_I as this system is defined over $\mathbb{F}_{q^m}$. Hence, let $J \subset \{n_1+1,\dots,n_2\}$ having cardinality $\#J = d_I$. Denote by $M_R[:,J]$ the submatrix of M_R consisting of the columns indexed by J. We only require $M_R[:,J]$ to have rank d_I, which holds for most choices of J, so typically we take the last d_I columns. This makes explicit that the system is underdetermined, and in case $M_R[:,J]$ has rank lower than d_I, we can include further columns. From (4) we can compute P_I by solving

$$\pi(\Psi(P_I(x)))M_R[:,J] = \mathbf{0} \tag{5}$$

Since also $\lambda\pi(\Psi(P_I(x)))M_R[:,J] = \mathbf{0}$ for any constant $\lambda \in \mathbb{F}_{q^m}$ we can recover P_I only up to such a constant factor. We will now show that this is not a problem. Let $P_I'(x) = \lambda P_I(x)$. We can recover $P_O'(x) = P_P(x)/P_I'(x) = P_O(x)/\lambda$, then $\mathbf{z}'(x) = P_H(x)/P_O'(x) = P_I'(x)\mathbf{x}(x)^{-1}\mathbf{y}(x) = \lambda P_I(x)\mathbf{x}(x)^{-1}\mathbf{y}(x)$, and finally $\mathbf{z}'(x)/P_I'(x) = \mathbf{x}(x)^{-1}\mathbf{y}(x)$ which corresponds to a ROLLO-I public key.

Similarly, for the ciphertext, we can recover

$$\lambda\mathbf{c}''(x) = \mathbf{c}(x)/P_O'(x) = \lambda\Psi(P_I(x))P_{E_1}(x) + \lambda\Psi(\mathbf{z}(x))P_{E_2}(x) \bmod P_2(x).$$

Since λ is constant, the degree of the right-hand side is below n_2 and we can reduce modulo P_1 and divide by $\lambda P_I(x)$ to get $P_{E_1}(x) + \lambda\mathbf{x}(x)^{-1}\mathbf{y}(x)P_{E_2}(x)$, matching the ROLLO-I ciphertexts. Note that the degree constraint on $\deg(P_{E_i})$ bounding it from above by $n_2 - n_1 - d_I$ for all proposed parameters implies that $\deg(P_{E_i}) < n_1$, hence, this is a valid ROLLO-I ciphertext. While this is not pointed out in [7], this is also required for the ROLLO-I decoder to work as $\mathsf{RSR}(\langle\mathbf{x},\mathbf{y}\rangle, \mathbf{x}\mathbf{c}', r)$ in the regular decapsulation procedure.

Estimates for the security of MLR1 The proposed parameters for MLR1 along with the attack costs are displayed in Table 2. The complexities of the attacks are computed using the script provided in [11], the Sage script performs puncturing of the public code to find the optimal complexity. For each security level, the costs of the attacks on the proposed parameters are shown in the third column of Table 2, and those of reduced Layered ROLLO-I along are in the fourth column of Table 2. The time in seconds to compute the public key transformation described in this section, on a Linux Mint virtual machine, is stated in the fifth column of Table 2. Furthermore, Table 2 shows that the security is still lower for these parameters than the targeted security levels, even though the designers were now aware of the attacks in [4].

Note that here we use P_I with $\deg P_I = d_I$ as stated in [7]. The parameters file in the implementation package instead uses $\deg P_I = 4$ for all security levels.

Security	$(q, n_1, n_2, d_I, m, r, d)$	Cost	Cost red.	Time (s)
128	$(2, 37, 61, 11, 67, 6, 2)$	103.83 [3]	96.95 [3]	1.85
192	$(2, 43, 71, 15, 79, 7, 3)$	185.52 [2]	156.16 [3]	2.42
256	$(2, 53, 103, 20, 97, 7, 3)$	187.91 [3]	151.11 [3]	4.21

Table 2. Values of the $\log_2$ of attack costs for MLR1's suggested parameters, before and after our reduction, and time consumed by the reduction. References after the costs refer to the publication which has the best attack on these parameters.

4 Message Recovery Attacks

We describe the two message recovery attacks that we mounted against Layered ROLLO-I. The first one breaks all the versions described so far and MLR2 (see Section 4.1). The second one applies to security levels 128 and 192 of MLR3 (see Section 4.3). The idea is to reduce the modular equation in the encapsulation to a system of linear equations and exploit the knowledge of zero positions of the error vectors to solve the system.

In the following, we first describe another modification MLR2 the designers made which changes the structure of the public key to counter the attacks described in the previous section. This and both previous versions use the same equation for encapsulation, albeit with different constraints on the degrees of the P_{E_i}. The message recovery attack works solely with this equation and thus applies to all these versions, hence we put the attack after the description of MLR2 to demonstrate the range of applicability, but want to stress that it applies already to the journal version [10] and is not a byproduct of the designers' patches. Finally, we describe and then attack a third modified version MLR3.

4.1 Second Modified Layered ROLLO-I (MLR2)

In this subsection, we describe the system from [8]. The new version of Layered ROLLO-I, which we denote by MLR2, uses polynomial masking techniques in order to avoid the reduction to ROLLO-I described in Section 3.2. To this end, the new system patch introduces an auxiliary polynomial P_N of small degree and modifies the P_P-part of the public key.

The values $(q, n_1, n_2, n_I, m, r, d)$, where $n_I < n_1 < n_2$ are the system parameters. There is also a primitive polynomial P_2 of degree n_2 which is a system parameter. We will report here only the key generation procedure, as the rest is the same as for MLR1 except for the degree of the error polynomials. The key generation procedure of the new system works as follows.

- KeyGen:
  - Pick random $\mathbf{x}, \mathbf{y} \in S_d^{n_1}(\mathbb{F}_{q^m})$.
  - Pick random primitive $P_1(x) \in \mathbb{F}_{q^m}[x]$ of degree n_1.
  - Pick random $P_I(x) \in \mathbb{F}_{q^m}[x]/(P_1(x))$ of degree n_I.
  - Pick random $P_O(x), P_N(x) \in \mathbb{F}_{q^m}[x]/(P_2(x))$, with $\deg P_N = n_N$.

- Set $\mathbf{z}(x) = P_I(x)\mathbf{x}(x)^{-1}\mathbf{y}(x) \bmod P_1(x)$.
- Set $P_P(x) = P_O(x)(\Psi(P_I(x)) + P_N(x)P_1(x)) \bmod P_2(x)$ and $P_H(x) = P_O(x)\Psi(\mathbf{z}(x)) \bmod P_2(x)$.
- Return $\mathsf{pk} = (P_P, P_H)$ and $\mathsf{sk} = (\mathbf{x}, \mathbf{y}, P_O, P_I, P_1)$.

The encapsulation mechanism with updated error weights is equivalent to that of MLR1 except that the random vectors $\mathbf{e}_1, \mathbf{e}_2$ should each correspond to a polynomial of degree $n_E < n_2 - n_1 - n_I - n_N - 2$.

4.2 Message recovery attack on all versions described so far

Recall that encapsulation for all versions of Layered ROLLO-I computes the ciphertext as

$$\mathbf{c}(x) = P_{E_1}(x)P_P(x) + P_{E_2}(x)P_H(x) \bmod P_2(x),$$

where we put $P_2 = P^b$ for the first version to unify notation. The public key is $\mathsf{pk} = (P_P, P_H)$ and the degree of the error polynomials P_{E_i} is limited to n_E to permit decapsulation.

Using the public key, we can compute $P_H(x)^{-1} \bmod P_2(x)$ and multiply the ciphertext polynomial by it to obtain

$$\bar{\mathbf{c}}(x) = \mathbf{c}(x)P_H(x)^{-1} = P_{E_1}(x)R(x) + P_{E_2}(x) \bmod P_2(x), \tag{6}$$

where $R(x) = P_P(x)P_H(x)^{-1} \bmod P_2(x)$. View equation (6) in terms of $\mathbb{F}_{q^m}$ vectors corresponding to the coefficient vectors of the polynomials involved. As in Section 3.2, we can regard R as the $(n_E+1)\times n_2$ full rank matrix M_R over $\mathbb{F}_{q^m}$ representing the multiplication of a polynomial of degree up to n_E by R modulo P_2, defined as in (3). In other words, M_R generates a linear $[n_2, n_E + 1]$-code over $\mathbb{F}_{q^m}$.

With this in mind we can rewrite (6) as

$$\bar{\mathbf{c}} = \mathbf{e}_1 M_R + \mathbf{e}_2, \tag{7}$$

which corresponds to a McEliece-like encryption of the "message" $\mathbf{e}_1$ and using $\mathbf{e}_2$ as error vector. Due to the low degree of the polynomial P_{E_2}, the error vector $\mathbf{e}_2$ also has a relatively low Hamming weight and we could use information-set decoding attacks to recover $\mathbf{e}_1$ and $\mathbf{e}_2$.

However, in the settings of all the versions of Layered ROLLO-I we have much more information on $\mathbf{e}_2$: We can exploit the low degree of the polynomial P_{E_2}, meaning that we know that the top $n_2 - n_E - 1$ positions are 0. (See Section 2 for the conversion between polynomials and vectors.) Indeed, we can find an invertible submatrix of M_R that consists of a subset of columns corresponding to error-free positions in the ciphertext by searching for an invertible submatrix of $M_R[:, n_E + 2 : n_2]$. Picking $n_E + 1$ random columns of a rank $n_E + 1$ matrix over $\mathbb{F}_{q^m}$, where q and m are given by the suggested parameters, will constitute an invertible matrix with overwhelming probability. We can also just take the last $n_E + 1$ columns. Let $M_{R\mathsf{inv}}$ be such a matrix. The last step is to compute $\mathbf{e}_1 = \bar{\mathbf{c}}' M_{R\mathsf{inv}}^{-1}$, where $\bar{\mathbf{c}}'$ consists of the coordinates of $\bar{\mathbf{c}}$ corresponding to the columns of $M_{R\mathsf{inv}}$. Finally, compute $\mathbf{e}_2 = \bar{\mathbf{c}}[1 : n_E + 1] - \mathbf{e}_1 M_R[:, 1 : n_E + 1]$.

We implemented this attack in SageMath. An average of the time required, on a Linux Mint virtual machine, to recover the plaintext for the proposed parameters of MLR2 is given in Table 3.

Table 3. Average time in seconds (on 50 samples for each security level) needed to recover a plaintext.

Security	n_E	Time (s)
128	17	2.21
192	19	3.18
256	39	6.65

Remark 2. We would like to remark that this message recovery attack works for all three versions of Layered ROLLO-I presented to so far since the degrees of $\mathbf{e}_1$ and $\mathbf{e}_2$ are smaller than half of n_2, which is relevant for the positions in $M_{R\mathsf{inv}}$ not to overlap with the positions in $\mathbf{e}_2$.

4.3 Third Modified Layered ROLLO-I (MLR3)

In this subsection, we describe the system from [9]. MLR3 uses polynomial masking in the ciphertext to overcome the message recovery attack that we described in the previous subsection. We will only display the parts in the specification of KeyGen and Encap that differ from that of MLR2. The values $(q, n_1, n_2, n_I, n_A, m, r, d)$, where $n_I = n_1 < n_2$ and $n_A = 4$ are the system parameters. The updates to the key generation procedure of the new system are as follows.

- KeyGen:
  - Pick random $P_{N,A}(x), P_{N,B}(x) \in \mathbb{F}_{q^m}[x]/(P_2(x))$ of degree n_A
  - Set
    $P_P(x) = P_O(x)(\Psi(P_I(x)) + P_{N,A}(x)P_1(x)) \bmod P_2(x)$,
    $P_H(x) = P_O(x)\Psi(\mathbf{z}(x)) \bmod P_2(x)$, and
    $P_B(x) = P_O(x)P_{N,B}(x)P_1(x) \bmod P_2(x)$.
  - Return $\mathsf{pk} = (P_P, P_H, P_B)$ and $\mathsf{sk} = (\mathbf{x}, \mathbf{y}, P_O, P_I, P_1)$.

The updates to the encapsulation mechanism with updated error weights are as follows.

- Encap(pk):
  - Compute
    $\mathbf{c}(x) = P_P(x)P_{E_1}(x) + P_H(x)P_{E_2}(x) + P_B(x)P_{N,C}(x) \bmod P_2(x)$,

where P_{E_1}, P_{E_2} and $P_{N,C}$ have degree $n_E < n_2 - n_1 - n_A - 1$. The decapsulation procedure has not been updated.

Message recovery attack on MLR3 We describe a fast message recovery attack on the security levels 128 and 192 of MLR3, that uses only linear algebra.

Compute the polynomials

$$\begin{aligned} A_1(x) &= P_P(x)P_B^{-1}(x),\ B_1(x) = P_H(x)P_B^{-1}(x), \\ A_2(x) &= P_P(x)P_H^{-1}(x),\ C_2(x) = P_B(x)P_H^{-1}(x),\ \text{and} \\ B_3(x) &= P_H(x)P_P^{-1}(x),\ C_3(x) = P_B(x)P_P^{-1}(x), \end{aligned} \tag{8}$$

and let $M_{A_1}, M_{B_1}, M_{A_2}, M_{C_2}, M_{B_3}$ and M_{C_3} be the corresponding matrices as in (3). Set

$$\begin{aligned} \mathbf{c}_1(x) &= \mathbf{c}(x)P_B^{-1}(x), \\ \mathbf{c}_2(x) &= \mathbf{c}(x)P_H^{-1}(x),\ \text{and} \\ \mathbf{c}_3(x) &= \mathbf{c}(x)P_P^{-1}(x). \end{aligned} \tag{9}$$

From these values, we derive the following equations

$$\begin{aligned} \mathbf{c}_1 &= \mathbf{e}_1 M_{A_1} + M_{B_1}\mathbf{e}_2 + \mathbf{p}, \\ \mathbf{c}_2 &= \mathbf{e}_1 M_{A_2} + \mathbf{e}_2 + \mathbf{p}M_{C_2},\ \text{and} \\ \mathbf{c}_3 &= \mathbf{e}_1 + \mathbf{e}_2 M_{B_3} + \mathbf{p}M_{C_3}, \end{aligned} \tag{10}$$

where we denote the coefficient vector of $P_{N,C}$ by $\mathbf{p}$. A first key observation is that, if we restrict to the last $n_2 - \ell$ columns of each matrix, corresponding to the terms of degree $\geq \ell$, we can remove the terms $\mathbf{p}, \mathbf{e}_2$ and $\mathbf{e}_1$ from the first, second and third equation in (10), respectively. A second key observation is that, thanks to the size of the field $\mathbb{F}_{q^m}$, we can find three sets $S_1, S_2, S_3 \subset [\ell+1, n_2]$ of cardinality ℓ such that the matrices $\overline{M}_{A_1} = M_{A_1}[:, S_1], \overline{M}_{B_1} = M_{B_1}[:, S_1]$, coinciding on S_1, the matrices $\overline{M}_{A_2} = M_{A_2}[:, S_2], \overline{M}_{C_2} = M_{C_2}[:, S_2]$, coinciding on S_2, and the matrices $\overline{M}_{B_3} = M_{B_3}[:, S_3], \overline{M}_{C_3} = M_{C_3}[:, S_3]$, coinciding on S_3, are all invertible $\ell \times \ell$ matrices. Denote by $\overline{\mathbf{c}}_1, \overline{\mathbf{c}}_2$ and $\overline{\mathbf{c}}_3$ the subvectors of $\mathbf{c}_1, \mathbf{c}_2$ and $\mathbf{c}_3$ of consisting of entries indexed by S_1, S_2 and S_3, respectively.

$$\begin{aligned} \overline{\mathbf{c}}_1 &= \mathbf{e}_1\overline{M}_{A_1} + \mathbf{e}_2\overline{M}_{B_1}, \\ \overline{\mathbf{c}}_2 &= \mathbf{e}_1\overline{M}_{A_2} + \mathbf{p}\overline{M}_{C_2},\ \text{and} \\ \overline{\mathbf{c}}_3 &= \mathbf{e}_2\overline{M}_{B_3} + \mathbf{p}\overline{M}_{C_3}. \end{aligned} \tag{11}$$

Let

$$\begin{aligned} M_{\mathbf{p}} &= \overline{M}_{C_2}\overline{M}_{A_2}^{-1}\overline{M}_{A_1}\overline{M}_{B_1}^{-1} + \overline{M}_{C_3}\overline{M}_{B_3}^{-1}\ \text{and} \\ \mathbf{c}_{\mathbf{p}} &= \overline{\mathbf{c}}_2\overline{M}_{A_2}^{-1}\overline{M}_{A_1}\overline{M}_{B_1}^{-1} - \overline{\mathbf{c}}_1\overline{M}_{B_1}^{-1} + \overline{\mathbf{c}}_3\overline{M}_{B_3}^{-1}, \end{aligned} \tag{12}$$

and observe that this simplifies to the linear system of equations

$$\mathbf{p}M_{\mathbf{p}} = \mathbf{c}_{\mathbf{p}},$$

which we can solve for $\mathbf{p}$. Substituting $\mathbf{p}$ into (11) we recover $\mathbf{e}_1$ and $\mathbf{e}_2$.

We implemented this attack in SageMath. An average of the time required, on a Linux Mint virtual machine, to recover the plaintext for the proposed parameters is given in Table 4.

Security	n_E	Time (s)
128	17	11.66
192	21	16.32

Table 4. Average time in seconds (on 50 samples for each security level) needed to recover a plaintext.

Remark 3. For the parameters of any security level, we always have that $3(n - \ell) > n_2$ where there exist at most n_2 linearly independent equations. For levels 128 and 192, we have $3\ell < n_2$ ensuring a unique solution of the system, which is not the case for security level 256.

Acknowledgments

The authors would like to thank Daniel J. Bernstein for discussions regarding recovering P_I in the attack presented in Section 3.2.

References

1. Aragon, N., Blazy, O., Deneuville, J.C., Gaborit, P., Hauteville, A., Ruatta, O., Tillich, J.P., Zémor, G., Aguilar Melchor, C., Bettaieb, S., Bidoux, L., Bardet, M., Otmani, A.: ROLLO. Tech. rep., NIST (2019), available at Round 2 page
2. Bardet, M., Briaud, P., Bros, M., Gaborit, P., Neiger, V., Ruatta, O., Tillich, J.P.: An algebraic attack on rank metric code-based cryptosystems. In: Eurocrypt 2020. LNCS, vol. 12107, pp. 64–93 (2020). https://doi.org/10.1007/978-3-030-45727-3_3
3. Bardet, M., Briaud, P., Bros, M., Gaborit, P., Tillich, J.P.: Revisiting algebraic attacks on MinRank and on the rank decoding problem. Designs, Codes and Cryptography pp. 1–37 (2023). https://doi.org/10.1007/s10623-023-01265-x
4. Bardet, M., Bros, M., Cabarcas, D., Gaborit, P., Perlner, R.A., Smith-Tone, D., Tillich, J.P., Verbel, J.A.: Improvements of algebraic attacks for solving the rank decoding and MinRank problems. In: Asiacrypt 2020. vol. 12491, pp. 507–536 (2020). https://doi.org/10.1007/978-3-030-64837-4_17
5. Gabidulin, E.M., Paramonov, A.V., Tretjakov, O.V.: Ideals over a non-commutative ring and thier applications in cryptology. In: Eurocrypt 1991. LNCS, vol. 547, pp. 482–489. Springer (1991)

6. Kim, C., Kim, Y.S., No, J.S.: Layered ROLLO-I. Submission to KpqC Competition Round 1 (2022)
7. Kim, C., Kim, Y., No, J.: Comments and modification on Layered ROLLO on kPQC-forum. Slides attached to reply on Kpqc bulletin (2023)
8. Kim, C., Kim, Y., No, J.: Comments and modification on Layered ROLLO on kPQC-forum. Slides attached on KpqC Bulletin (2023)
9. Kim, C., Kim, Y., No, J.: Comments and modification on Layered ROLLO on kPQC-forum. Slides attached to reply on KpqC Bulletin (2023)
10. Kim, C., Kim, Y., No, J.: New design of blockwise interleaved ideal low-rank parity-check codes for fast post-quantum cryptography. IEEE Commun. Lett. **27**(5), 1277–1281 (2023)
11. Lange, T., Pellegrini, A., Ravagnani, A.: On the security of REDOG. Cryptology ePrint Archive, Paper 2023/1205 (2023)
12. Overbeck, R.: Structural attacks for public key cryptosystems based on Gabidulin codes. Journal of Cryptology **21**(2), 280–301 (2008)