Breaking 2/3 of Layered-ROLLO-I-2023-10-20

Alex Pellegrini[1]

[1]Eindhoven University of Technology

October 22, 2023

Abstract

In this short note we describe a full break for security level 128 and 192 of the Layered-ROLLO-I cryptosystem submitted to the Korean post-quantum cryptography competition. We show that it is possible to quickly compute the plaintext from a ciphertext given the public data and using only linear algebra. Our attack takes few seconds to decrypt ciphertexts for both security levels on a Linux Mint virtual machine.

1 The new Layered-ROLLO-I

In this section we describe the system from [KKN23]. The once more patched version of Layered ROLLO-I, now up to version 4, uses polynomial masking techniques on the public key in order to avoid the reduction to ROLLO-I described in [LP23]. The same masking technique has been applied to the ciphertext in order to avoid the message recovery attack proposed in [Pel23]. To this end, the new system patch introduces auxiliary polynomials $P_{N,A}, P_{N,B}$ and $P_{N,C}$ of small degree and adds the P_B-part of the public key. This technique successfully overcomes the attack Pellegrini sets up.

The following text uses notation as in [LP23, Pel23], please see there for definitions. The values $(q, n_1, n_2, n_I, n_A, m, r, d)$, 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.

The updated 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 $P_1 \in \mathbb{F}_{q^m}[x]$ of degree n_1.
- Pick random $P_I \in \mathbb{F}_{q^m}[x]/(P_1)$ of degree n_I.
- Pick random $P_O \in \mathbb{F}_{q^m}[x]/(P_2)$.
- Pick random $P_{N,A}, P_{N,B} \in \mathbb{F}_{q^m}[x]/(P_2)$ of degree n_A

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

Remark 1.1 *Note that in the new specification P_1 is claimed to be part of the secret key, while in the reference implementation it is still a **fixed** polynomial. See* `Layered_ROLLO_Gen-0922/src/schemes/biix/biix_kem.c`*, lines 62-64.*

As described in the reference implementation of the new patch, precisely in line 491 of `Layered_ROLLO_Gen-0922/src/schemes/biix/biix_kem.c` the error vectors are represented by an $\mathbb{F}_{q^m}$-tuple of $\ell := n_2 - n_1 - n_A - 1$ elements (coefficients vectors). The error vectors can equivalently be represented as polynomials in $\mathbb{F}_{q^m}[x]$ of degree at most $n_E = \ell - 1$.

Remark 1.2 *When viewing a polynomial $P \in \mathbb{F}_{q^m}[x]$ of degree at most n as an element of $v \in \mathbb{F}_{q^m}^{n+1}$ corresponding to its coefficient vector, we consider the entries of v to be ordered in a way such that $P = \sum_{i=0}^{n} v_i x^i$.*

The updated encapsulation mechanism with updated error-weights works as follows.

Encap(pk):

- Pick random $E = \langle \mathbf{e}_1, \mathbf{e}_2 \rangle \in S_r^{n_2}(\mathbb{F}_{q^m})$, with $\mathbf{e}_1, \mathbf{e}_2$ each corresponding to a polynomial of degree $n_E < n_2 - n_1 - n_A - 1$.
- Pick random $P_{N,C} \in \mathbb{F}_{q^m}[x]$ of degree n_E
- Set $P_{E_1} = \mathbf{e}_1(x)$ and $P_{E_2} = \mathbf{e}_2(x)$.
- Compute $\mathbf{c}(x) = P_P P_{E_1} + P_H P_{E_2} + P_B P_{N,C} \bmod P_2$.
- Return $K = \mathsf{hash}(E)$ and $\mathbf{c}$.

The decapsulation procedure has not been updated, we include it only for completeness.

Decap(sk):

- Compute $\mathbf{c}'' = P_O^{-1}\mathbf{c}(x) \bmod P_2$.
- Compute $\mathbf{c}' = P_I^{-1}\Omega(\mathbf{c}'') \bmod P_1$.
- Decode $E = \mathsf{RSR}(\langle \mathbf{x}, \mathbf{y} \rangle, \mathbf{x}\mathbf{c}', r)$.
- Return $K = \mathsf{hash}(E)$.

Finally, the suggested parameters for the new modified Layered-ROLLO-I cryptosystem are

Security parameter	$(q, m, n_I, n_1, n_2, n_A)$
128	$(2, 67, 37, 37, 61, 4)$
192	$(2, 79, 43, 43, 71, 4)$
256	$(2, 97, 53, 53, 103, 4)$

Table 1: Suggested parameters for the new version.

2 The message recovery attack

This section describes the message recovery attack that we mounted against the Layered-ROLLO-I cryptosystem. The proposed strategy only uses linear algebra to recover $\mathbf{e}_1, \mathbf{e}_2$ and $P_{N,C}$. We will compute initial data from the public key and use it to describe linear relations among the three pieces of the ciphertext.

Recall that encapsulation computes the ciphertext

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

. From the public key we can compute the three tuples of values

$$\begin{aligned} T_1 &= (A_1 = P_P P_B^{-1}, B_1 = P_H P_B^{-1}, \mathbf{c}_1(x) = \mathbf{c}(x) P_B^{-1}) \\ T_2 &= (A_2 = P_P P_H^{-1}, C_2 = P_B P_H^{-1}, \mathbf{c}_2(x) = \mathbf{c}(x) P_H^{-1}) \\ T_3 &= (B_3 = P_H P_P^{-1}, C_3 = P_B P_P^{-1}, \mathbf{c}_3(x) = \mathbf{c}(x) P_P^{-1}) \end{aligned} \tag{1}$$

Each tuple T_i with $i = 1, 2, 3$ represents an equation as follows

$$\begin{aligned} \mathbf{c}_1(x) &= A_1 P_{E_1} + B_1 P_{E_2} + P_{N,C} \\ \mathbf{c}_2(x) &= A_2 P_{E_1} + P_{E_2} + C_2 P_{N,C} \\ \mathbf{c}_3(x) &= P_{E_1} + B_3 P_{E_2} + C_3 P_{N,C}. \end{aligned} \tag{2}$$

For each of the values A_1, B_1, A_2, C_2, B_3 and C_3 we can compute a matrix over $\mathbb{F}_{q^m}$ representing the multiplication of a polynomial of degree up to n_E by each value modulo P_2, e.g. for A_1 we compute

$$M_{A_1} = \begin{pmatrix} A_1 \bmod P_2 \\ A_1 x \bmod P_2 \\ \vdots \\ A_1 x^{n_E} \bmod P_2 \end{pmatrix},$$

where each row consists of the coefficient vector of $A_1 x^i \bmod P_2$ for $i = 0, \ldots, n_E$. We compute all the matrices $M_{A_1}, M_{B_1}, M_{A_2}, M_{C_2}, M_{B_3}$ and M_{C_3}. View equation (2) in terms of $\mathbb{F}_{q^m}$ vectors corresponding to the coefficient vectors of the polynomials involved and rewrite as

$$\begin{aligned} \mathbf{c}_1 &= \mathbf{e}_1 M_{A_1} + M_{B_1} \mathbf{e}_2 + \mathbf{p} \\ \mathbf{c}_2 &= \mathbf{e}_1 M_{A_2} + \mathbf{e}_2 + \mathbf{p} M_{C_2} \\ \mathbf{c}_3 &= \mathbf{e}_1 + \mathbf{e}_2 M_{B_3} + \mathbf{p} M_{C_3} \end{aligned} \tag{3}$$

where we denote the coefficient vector of $P_{N,C}$ with $\mathbf{p}$. 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 obtain

$$\mathbf{c}_1[:, \ell : n_2] = \mathbf{e}_1 M_{A_1}[:, \ell : n_2] + \mathbf{e}_2 M_{B_1}[:, \ell : n_2] \tag{4}$$
$$\mathbf{c}_2[:, \ell : n_2] = \mathbf{e}_1 M_{A_2}[:, \ell : n_2] + \mathbf{p} M_{C_2}[:, \ell : n_2] \tag{5}$$
$$\mathbf{c}_3[:, \ell : n_2] = \mathbf{e}_2 M_{B_3}[:, \ell : n_2] + \mathbf{p} M_{C_3}[:, \ell : n_2], \tag{6}$$

getting rid of one of the terms in each equation. 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 $\overline{M}_{A_1} = M_{A_1}[:, S_1], \overline{M}_{B_1} = M_{B_1}[:, S_1], \overline{M}_{A_2} = M_{A_2}[:, S_2], \overline{M}_{C_2} = M_{C_2}[:, S_2], \overline{M}_{B_3} = M_{B_3}[:, S_3], \overline{M}_{C_3} = M_{C_3}[:, S_3]$ are all invertible $\ell \times \ell$ matrices. Denote also $\overline{\mathbf{c}}_1, \overline{\mathbf{c}}_2$ 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. Replacing into (4),(5) and (6) we have that the equalities still hold. Moreover, from (4) we get

$$\mathbf{e}_1 = (\overline{\mathbf{c}}_1 - \mathbf{e}_2 \overline{M}_{B_1}) \overline{M}_{A_1}^{-1}, \tag{7}$$

while from (5)

$$\mathbf{e}_1 = (\overline{\mathbf{c}}_2 - \mathbf{p} \overline{M}_{C_2}) \overline{M}_{A_2}^{-1}. \tag{8}$$

Combining (7) and (8) we can express $\mathbf{e}_2$ in terms of $\mathbf{p}$ as

$$\mathbf{e}_2 = \mathbf{p} \overline{M}_{C_2} \overline{M}_{A_2}^{-1} \overline{M}_{A_1} \overline{M}_{B_1}^{-1} - \overline{\mathbf{c}}_2 \overline{M}_{A_2}^{-1} \overline{M}_{A_1} \overline{M}_{B_1}^{-1} + \overline{\mathbf{c}}_1 \overline{M}_{B_1}^{-1} \tag{9}$$

From (6) we compute

$$\mathbf{e}_2 = (\overline{\mathbf{c}}_3 - \mathbf{p} \overline{M}_{C_3}) \overline{M}_{B_3}^{-1}. \tag{10}$$

Combining (9) and (10) we end up with a system of ℓ linear equations that allows us to compute $\mathbf{p}$ from public data only. Formally

$$\mathbf{p}(\overline{M}_{C_2} \overline{M}_{A_2}^{-1} \overline{M}_{A_1} \overline{M}_{B_1}^{-1} + \overline{M}_{C_3} \overline{M}_{B_3}^{-1}) = \overline{\mathbf{c}}_2 \overline{M}_{A_2}^{-1} \overline{M}_{A_1} \overline{M}_{B_1}^{-1} - \overline{\mathbf{c}}_1 \overline{M}_{B_1}^{-1} + \overline{\mathbf{c}}_3 \overline{M}_{B_3}^{-1}. \tag{11}$$

Once $\mathbf{p}$ has been recovered we can simply plug it into (8) and (9) to obtain $\mathbf{e}_1$ and $\mathbf{e}_2$ recovering the entire plaintext.

We implemented a non-optimized version of our 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 the following table. It is worth noting that in our experiments we always used error vectors of the maximum allowed Hamming weight in order to simulate the worst case scenario for our attack.

Security level	max degree of $\mathbf{e}_1$ and $\mathbf{e}_2$	Time (s)
128	18	11.66
192	22	16.32

Table 2: Average time in seconds (on 50 samples for each security level) needed to recover a plaintext.

Remark 2.1 *The author wants to thank Tanja Lange for pointing out the following shortcut that doesn't require the second key observation. It is indeed possible to just regard equations* (7),(8) *and* (9) *as a system of* $3(n-\ell)$ *linear equations in* 3ℓ *unknowns. 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. Now, for level 128 and 192 we have* $3\ell < n_2$ *ensuring unique solution of the system, which is not the case for security level 256. The method explained in this writeup is in any case faster as it deals with smaller systems of equations and uses the structure of the resulting matrix which would have large blocks of* $0s$*.*

References

[KKN23] Chanki Kim, Young-Sik Kim, and Jong-Seon No. Comments and modification on Layered ROLLO on kPQC-forum. Slides attached to reply on Kpqc Bulletin, 2023.

[LP23] Tanja Lange and Alex Pellegrini. Analysis of Layered ROLLO-I. Email to KpqC Bulletin, 2023.

[Pel23] Alex Pellegrini. An efficient message recovery attack on the new Modified-Layered-ROLLO-I. Email to KpqC Bulletin, 2023.