A Reaction Attack on LEDApkc

Tomáš Fabšič [⋆] [⋆⋆], Viliam Hromada [⋆] [⋆⋆], Pavol Zajac [⋆] [⋆⋆]

Slovak University of Technology in Bratislava
Faculty of Electrical Engineering and Information Technology
Ilkovičova 3, 81219 Bratislava, Slovak Republic
{tomas.fabsic, viliam.hromada, pavol.zajac}@stuba.sk

Abstract. We propose a new reaction attack on the public-key cryptosystem LEDApkc. The adversary uses the decoding failure rate (DFR) analysis to learn information about the secret masking matrix Q. Provided the adversary learns information about Q within $10^4 \times \mathrm{DFR}^{-1}$ decryptions (as prescribed by LEDApkc design to thwart previously known attacks), the adversary builds a small set of candidates for Q. Using these candidates, the adversary obtains candidates for a generator matrix of the secret LDPC code. Afterwards, the adversary applies Stern's algorithm to recover the secret matrix H, thus recovering the full private key.

Provided the adversary can learn information about the matrix Q, the complexity of the attack is below 2^{99} for a parameter set for 128-bit security. In order to study whether the adversary can learn information about Q from $10^4 \times \mathrm{DFR}^{-1}$ decryptions, we conducted experiments with a modified parameter set. The parameter set was modified only in order to increase the DFR, and thus make experiments less computationally expensive. We show that with the modified parameter set it is indeed possible to learn the required information about the matrix Q.

Keywords: LEDApkc, QC-LDPC McEliece cryptosystem, reaction attack, post-quantum cryptography

1 Introduction

LEDApkc, [1], is a public-key cryptosystem recently submitted to NIST's Post-Quantum Cryptography Standardization Process. It is very similar to the QC-LDPC McEliece cryptosystem, which was presented by Baldi and Chiaraluce in [2], and later amended in [3]. Following the work of Guo et al., [5], Fabšič et al. recently showed that a reaction attack can be mounted on the QC-LDPC

⋆ Support by grant VEGA 1/0159/17 is acknowledged.

⋆⋆ We are grateful to the HPC center at the Slovak University of Technology in Bratislava, which is a part of the Slovak Infrastructure of High Performance Computing (SIVVP project, ITMS code 26230120002, funded by the European region development funds, ERDF), for the computational time and resources made available.

McEliece cryptosystem [4]. The attack is facilitated by a dependence between secret matrices H and Q and the probability of the decoding failure (aka decoding failure rate, DFR) in the decryption algorithm. This dependence allows an adversary to learn information about the secret matrices H and Q, provided the adversary sends a large number of ciphertexts and learns whether they were successfully decrypted or not. In particular, the adversary learns the distances between the set bits (i.e. the bits equal to 1) in rows of H and in rows of Q. Similarly as in the QC-LDPC McEliece cryptosystem, the private key in LEDApkc contains matrices H and Q. To avoid the attack of Fabšič et al., the authors of LEDApkc propose to use a single key-pair in LEDApkc for at most $10^4 \times \mathrm{DFR}^{-1}$ decryptions.

In this paper we propose a reaction attack on LEDApkc, which differs from the previous reaction attack on QC-LDPC McEliece. The adversary in our attack is limited by the fact that he can send at most $10^4 \times \mathrm{DFR}^{-1}$ ciphertexts for decryption. After learning whether these ciphertexts were successfully decrypted, the adversary only tries to learn the distances in the matrix Q. The reason for this is that distances in Q are easier to distinguish than distances in H, as was demonstrated in [4]. If the adversary is able to learn the distances in Q, he can then build a set of candidates for Q. Using these candidates, the adversary obtains candidates for a generator matrix of the secret LDPC code. Afterwards, the adversary applies Stern's algorithm, [8], to find low-weight codewords in the dual of the secret LDPC code and thus recover the secret matrix H. The attack by Stern's algorithm was already considered in [2], and it was the reason for adding the matrix Q to the QC-LDPC McEliece cryptosystem.

Provided the adversary can learn the distances in the matrix Q, the complexity of the attack is below 2^{99} for a parameter set for 128-bit security. The authors of LEDApkc claim that for this parameter set the value of DFR is approximately 8.3×10^{-9}. In order to study whether the adversary can learn the distances in Q from $10^4 \times \mathrm{DFR}^{-1}$ decryptions we conducted experiments with a modified parameter set. The parameter set was modified in order to increase the DFR and thus make experiments less computationally expensive. We show that with a modified parameter set with the DFR of around 10^{-2} it is indeed possible to learn the required information about the matrix Q.

Our attack is relevant for cryptosystems based on QC-LDPC codes which use a sparse masking matrix Q with a small number of blocks. However, this attack is not applicable to similar QC-MDPC designs, [7], as they lack the extra masking matrix and vectors of small enough weight in the secret matrix H.

2 LEDApkc

For a detailed description of LEDApkc, we refer the reader to [1]. Here, we only highlight facts which are crucial for the attack.

The private key in LEDApkc consists of two matrices: H and Q. The matrix H is a sparse parity-check matrix for an LDPC code which is able to correct t

errors. Q is a square matrix and it is very sparse. Both H and Q are composed of circulant blocks.

The public key is formed by a matrix G'. G' is in the systematic form and it is again composed of circulant blocks. G' has the property that

$$G'(HQ)^T = 0. \tag{1}$$

The sender encrypts the plaintext u to obtain the ciphertext x as follows:

$$x = uG' + e, \tag{2}$$

where e is a randomly generated error vector with Hamming weight t.

To decrypt the message, the receiver uses a bitflipping algorithm which employes the matrices H and Q. The decryption fails with some probability. This probability is referred to as the decoding failure rate (DFR).

To achieve IND-CCA2 security, the authors of LEDApkc use the Kobara-Imai γ-conversion [6].

3 The Attack

To avoid the reaction attack presented in [4], the authors of LEDApkc propose to use a single key-pair for at most $10^4 \times \mathrm{DFR}^{-1}$ decryptions. In this section, we propose a different reaction attack. Our attack consists of the following steps:

1. The adversary sends at most $10^4 \times \mathrm{DFR}^{-1}$ ciphertexts and always observes whether the ciphertext was successfully decrypted or not.
2. Using this information, the adversary will try to learn the distances between the set bits in rows of Q. To learn the distances the adversary will proceed exactly as in [4].
3. Provided he succeeded to learn the distances, the adversary will build a set of candidates for Q (by the same method as in [4]).
4. Afterwards, the adversary will use the fact that $0 = G'(HQ)^T = (G'(Q^T))H^T$, which means that $G'(Q^T)$ is a generator matrix for the secret LDPC code. Thus the attacker will build a set of candidates for a generator matrix of the secret LDPC code.
5. The adversary will apply Stern's algorithm [8] on these candidates to find a low-weight codeword in the dual of the secret LDPC code and thus will recover the matrix H.

In the rest of this section, we will present the attack for the parameter set for 128-bit security with $n_0 = 2$ circulant blocks in H from [1]. This means that H will be of the form $[H_0|H_1]$, where H_0 and H_1 are circulant matrices of dimension $p \times p$, where $p = 27779$. Each column of H_0 and H_1 has Hamming weight equal to $d_v = 17$. The matrix Q has the form

$$Q = \begin{bmatrix} Q_1 & Q_2 \\ Q_3 & Q_4 \end{bmatrix},$$

where each Q_i is again a circulant matrix of dimension $p \times p$ ($p = 27779$). Each row of Q_1 and Q_4 has Hamming weight equal to $m_0 = 4$ and each row of Q_2 and Q_3 has Hamming weight equal to $m_1 = 3$. For these parameters, the authors of LEDApkc propose the value of t equal to 224 and they claim that the cryptosystem then has DFR equal to approximately 8.3×10^{-9}. In order to decrease computational costs of our experiments, we ran the experiments with modified values of t, resulting in higher values of DFR.

3.1 Learning Distances in Q

We define the distance between two ones in positions p_1 and p_2, $p_2 > p_1$, in a vector v of length p in the same way as in [5]:

$$d(p_1, p_2, p) = \min \{p_2 - p_1, p - (p_2 - p_1)\} .$$

Following [5], we call the set of all distances present in v the *distance spectrum* of v, and we denote it as $DS(v)$. Our definition implies that in a circulant matrix, every row has the same distance spectrum. Thus by a distance spectrum of a circulant matrix C (denoted as $DS(C)$) we will mean the distance spectrum of its first row.

We ran an experiment which is a variation of experiments in [5] and [4]. This experiment is presented in Algorithm 1. We used the reference implementation of LEDApkc submitted to NIST's Post-Quantum Cryptography Standardization Process. This implementation includes the Kobara-Imai γ-conversion [6]. In the implementation we increased the value of t to 254. We also increased the value of the parameter MAX_ENCODABLE_BIT_SIZE_CW_ENCODING to 2320, to avoid artificially high number of ones towards the end of the error vector e. With these changes, DFR increased to approximately 10^{-2}.

Algorithm 1

```
   INPUT: DFR
OUTPUT: vectors $a$, $b$, $y$ and $z$

1. $a \leftarrow$ zero-initialized vector of length $\lfloor p/2 \rfloor$
2. $b \leftarrow$ zero-initialized vector of length $\lfloor p/2 \rfloor$
3. $y \leftarrow$ zero-initialized vector of length $\lfloor p/2 \rfloor$
4. $z \leftarrow$ zero-initialized vector of length $\lfloor p/2 \rfloor$
5. $i \leftarrow 0$
6. while $i < 10^4 \times \mathrm{DFR}^{-1}$ do:
   (a) encrypt the zero vector using LEDApkc (we denote the output by $c$) and
       extract the error vector $e$ used during the encryption
   (b) divide $e$ as $e = (e^1|e^2)$, where each $e^i$ has length $p$
   (c) $s \leftarrow$ distances between ones in $e^1$
   (d) $r \leftarrow$ distances between ones in $e^2$
   (e) decrypt $c$ using LEDApkc
   (f) $l \leftarrow 1$ if the decoding failure occurs, 0 otherwise
```

```
(g) for $d$ from 1 to $\lfloor p/2 \rfloor$ do:
    i. if $s[d] \geq 1$ then:
        A. $a[d] \leftarrow a[d] + l$
        B. $b[d] \leftarrow b[d] + 1$
    ii. if $r[d] \geq 1$ then:
        A. $y[d] \leftarrow y[d] + l$
        B. $z[d] \leftarrow z[d] + 1$
(h) $i \leftarrow i + 1$
```

Algorithm 1 outputs four vectors: a, b, y, z. The value of $\frac{a[d]}{b[d]}$ represents the estimated probability of the decoding failure for ciphertexts with the property that the first half of the error vector contains distance d. Similarly, the value of $\frac{y[d]}{z[d]}$ represents the estimated probability of the decoding failure for ciphertexts with the property that the second half of the error vector contains distance d.

We can see the results of the experiment in Figure 1 and Figure 2. The results show that based on the values of $\frac{a[d]}{b[d]}$, one can distinguish which distances are present in blocks Q_1 and Q_3 (or equivalently in blocks Q_1^T and Q_3^T, since a circulant matrix and its transpose have the same distance spectrum). Similarly, the results show that based on the values of $\frac{y[d]}{z[d]}$, one can distinguish which distances are present in blocks Q_2 and Q_4 (or equivalently in blocks Q_2^T and Q_4^T). This is because during the decryption process the error vector e gets multiplied by Q^T. This means that the first half of e gets multiplied by ${Q_1}^T$ and ${Q_3}^T$ and the second half by ${Q_2}^T$ and ${Q_4}^T$. If the first half of e shares a distance with either Q_1 and Q_3, then the resulting product eQ^T has lower Hamming weight which leads to a lower probability of the decoding error. Similarly for the second half of e and blocks Q_2 and Q_4. A more detailed explanation of this phenomenon can be found in [4].

We conducted the same experiment with different values of t and we noticed one interesting phenomenon: As t decreased it was easier to distinguish the distances present in Q. Details are presented in Appendix. We conjecture that this phenomenon occurs due to the changing slope of the error-correction performance curve (the curve which shows the DFR of the decoder as a function of the number of errors in a codeword) of the decoder in LEDApkc.

3.2 Reconstruction of Q^T

Using the same method as in [4], the adversary reconstructs from the results in Figure 1 two pairs of vectors $\left\{\tilde{q}_1^1, \tilde{q}_1^2\right\}$ and $\left\{\tilde{q}_3^1, \tilde{q}_3^2\right\}$, such that one vector from the pair $\left\{\tilde{q}_1^1, \tilde{q}_1^2\right\}$ defines the block Q_1^T up to a cyclic shift and one vector from the pair $\left\{\tilde{q}_3^1, \tilde{q}_3^2\right\}$ defines the block Q_3^T up to a cyclic shift. Similarly, using the results in Figure 2, adversary reconstructs two pairs of vectors $\left\{\tilde{q}_2^1, \tilde{q}_2^2\right\}$ and $\left\{\tilde{q}_4^1, \tilde{q}_4^2\right\}$, such that one vector from the pair $\left\{\tilde{q}_2^1, \tilde{q}_2^2\right\}$ defines the block Q_2^T up to a cyclic shift and one vector from the pair $\left\{\tilde{q}_4^1, \tilde{q}_4^2\right\}$ defines the block Q_4^T up to a cyclic shift.

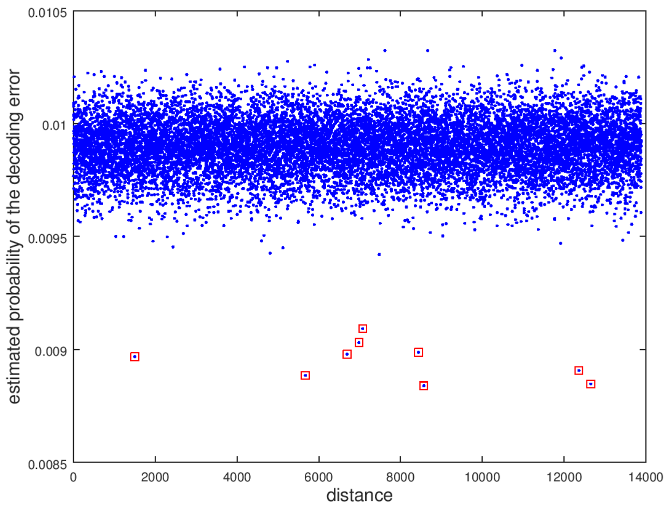

Fig. 1. Values of $\frac{a[d]}{b[d]}$ from the experiment presented in Algorithm 1. Values associated to distances present in one of the circulant blocks Q_1 and Q_3 are marked by red squares.

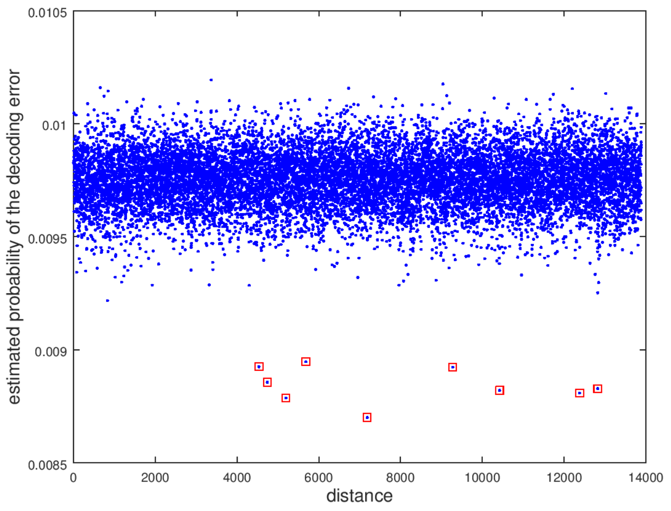

Fig. 2. Values of $\frac{y[d]}{z[d]}$ from the experiment presented in Algorithm 1. Values associated to distances present in one of the circulant blocks Q_2 and Q_4 are marked by red squares.

Let us express this in terms of polynomials. We will use the fact that the ring of circulant binary matrices of dimension $p \times p$ is isomorphic to the ring $\mathbb{Z}_2[x]/(x^p+1)$. The isomorphism maps a circulant matrix with the first row $(c_0, c_1, c_2, \ldots, c_{p-1})$ onto the polynomial $c(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_{p-1} x^{p-1}$. In addition, let $v(x)$ be the polynomial associated to the vector $v = (v_0, v_1, \ldots, v_{p-1})$ under the mapping $v \mapsto v_0 + v_1 x + v_2 x^2 + \cdots + v_{p-1} x^{p-1}$. Below, let us represent circulant blocks by their corresponding polynomials. Then the adversary knows that Q^T is of the form

$$Q^T = \left[\begin{array}{c|c} x^{k_1}\tilde{q}_1(x) & x^{k_3}\tilde{q}_3(x) \\ \hline x^{k_2}\tilde{q}_2(x) & x^{k_4}\tilde{q}_4(x) \end{array}\right],$$

for some $\tilde{q}_1 \in \{\tilde{q}_1^1, \tilde{q}_1^2\}$, $\tilde{q}_2 \in \{\tilde{q}_2^1, \tilde{q}_2^2\}$, $\tilde{q}_3 \in \{\tilde{q}_3^1, \tilde{q}_3^2\}$, $\tilde{q}_4 \in \{\tilde{q}_4^1, \tilde{q}_4^2\}$ and some $k_1, k_2, k_3, k_4 \in \{0, \ldots, p-1\}$.

3.3 Reconstruction of a Generator Matrix for the Secret LDPC Code

Afterwards, the adversary can build a set of candidates for the matrix $G = G'(Q^T)$. Since $0 = G'(HQ)^T = (G'(Q^T))H^T$, G is a generator matrix for the secret LDPC code. Since G' is in systematic form, we can represent it as $G' = [1|g(x)]$, $g(x) \in \mathbb{Z}_2[x]/(x^p+1)$. Thus we obtain that G is of the form

$$G = \left[\, x^{k_1}\tilde{q}_1(x) + x^{k_2}\tilde{q}_2(x)g(x) \middle| x^{k_3}\tilde{q}_3(x) + x^{k_4}\tilde{q}_4(x)g(x) \,\right], \tag{3}$$

for some $\tilde{q}_1 \in \{\tilde{q}_1^1, \tilde{q}_1^2\}$, $\tilde{q}_2 \in \{\tilde{q}_2^1, \tilde{q}_2^2\}$, $\tilde{q}_3 \in \{\tilde{q}_3^1, \tilde{q}_3^2\}$, $\tilde{q}_4 \in \{\tilde{q}_4^1, \tilde{q}_4^2\}$ and some $k_1, k_2, k_3, k_4 \in \{0, \ldots, p-1\}$.

If G given by Equation (3) is a generator matrix of the secret LDPC code, then so is the matrix $\tilde{G}$ given by

$$\tilde{G} = \left[\, \tilde{q}_1(x) + x^{k_2-k_1}\tilde{q}_2(x)g(x) \middle| x^{k_3-k_1}\tilde{q}_3(x) + x^{k_4-k_1}\tilde{q}_4(x)g(x) \,\right],$$

since it can be obtained from G by a row permutation. Thus, for the adversary it is enough to consider only matrices G of the form

$$G = \left[\, \tilde{q}_1(x) + x^{l_2}\tilde{q}_2(x)g(x) \middle| x^{l_3}\tilde{q}_3(x) + x^{l_4}\tilde{q}_4(x)g(x) \,\right], \tag{4}$$

for some $\tilde{q}_1 \in \{\tilde{q}_1^1, \tilde{q}_1^2\}$, $\tilde{q}_2 \in \{\tilde{q}_2^1, \tilde{q}_2^2\}$, $\tilde{q}_3 \in \{\tilde{q}_3^1, \tilde{q}_3^2\}$, $\tilde{q}_4 \in \{\tilde{q}_4^1, \tilde{q}_4^2\}$ and some $l_2, l_3, l_4 \in \{0, \ldots, p-1\}$.

3.4 Application of Stern's algorithm

After obtaining the set of candidates for the matrix G, the adversary will apply Stern's algorithm [8] on these candidates to find a codeword of weight $n_0 \times d_v$ in the dual of the secret LDPC code. Such word will allow the adversary to reconstruct the matrix H.

In fact, it suffices to apply Stern's algorithm on a subset of the set of candidates for G. Let v be a codeword of weight $n_0 \times d_v$ in the dual of the secret LDPC code. Let $v = (v^1|v^2)$, where each v^i has length p. Suppose that

$$G = \left[\, \tilde{q}_1(x) + x^{l_2}\tilde{q}_2(x)g(x) \middle| x^{l_3}\tilde{q}_3(x) + x^{l_4}\tilde{q}_4(x)g(x) \,\right]$$

is a generator matrix of the secret LDPC code. Then we have

$$\left[\, \tilde{q}_1(x) + x^{l_2}\tilde{q}_2(x)g(x) \middle| x^{l_3}\tilde{q}_3(x) + x^{l_4}\tilde{q}_4(x)g(x) \,\right] \begin{pmatrix} {v^1}^T \\ {v^2}^T \end{pmatrix} = 0,$$

which can be rewritten as

$$\left[\, \tilde{q}_1(x) + x^{l_2}\tilde{q}_2(x)g(x) \middle| \tilde{q}_3(x) + x^{l_4-l_3}\tilde{q}_4(x)g(x) \,\right] \left[\begin{array}{c|c} 1 & 0 \\ \hline 0 & x^{l_3} \end{array}\right] \begin{pmatrix} {v^1}^T \\ {v^2}^T \end{pmatrix} = 0.$$

Thus, the dual of the code generated by the matrix

$$\left[\tilde{q}_1(x) + x^{l_2}\tilde{q}_2(x)g(x) | \tilde{q}_3(x) + x^{l_4-l_3}\tilde{q}_4(x)g(x)\right]$$

contains the vector w, such that

$$w^T = \left[\begin{array}{c|c} 1 & 0 \\ \hline 0 & x^{l_3} \end{array}\right] \begin{pmatrix} {v^1}^T \\ {v^2}^T \end{pmatrix}.$$

The first half of w is equal to v^1, and the second half is equal to v^2 cyclically shifted by l_3 positions. This means that it suffices to apply Stern's algorithm only to matrices of the form

$$G = \left[\, \tilde{q}_1(x) + x^{j_2}\tilde{q}_2(x)g(x) \middle| \tilde{q}_3(x) + x^{j_4}\tilde{q}_4(x)g(x) \,\right], \tag{5}$$

for some $\tilde{q}_1 \in \{\tilde{q}_1^1, \tilde{q}_1^2\}$, $\tilde{q}_2 \in \{\tilde{q}_2^1, \tilde{q}_2^2\}$, $\tilde{q}_3 \in \{\tilde{q}_3^1, \tilde{q}_3^2\}$, $\tilde{q}_4 \in \{\tilde{q}_4^1, \tilde{q}_4^2\}$ and some $j_2, j_4 \in \{0, \ldots, p-1\}$. One of these matrices will then generate a code, dual of which will contain a vector w of weight $n_0 \times d_v$. Let $w = (w^1|w^2)$, where each w^i has length p. The secret matrix H then must have the form

$$H = [x^s w^1(x) | x^{s+p-l_3} w^2(x)], \tag{6}$$

for some $s \in \{0, \ldots, p-1\}$ and for the same l_3 as in Equation (4).

For the parameter set for 128-bit security with $n_0 = 2$ from [1], the adversary has to consider $2^4 \times 27779^2 < 2^{33.5235}$ different candidates for G. The adversary knows that one of these candidates generates a code, dual of which contain a vector w of weight $n_0 \times d_v$. The adversary will apply Stern's algorithm to candidates for G.

We consider Stern's algorithm with parameters $p = 3$ and $l = 48$, and base further results on the analysis of Stern's algorithm in [8]. If the algorithm is applied to the correct candidate for G, the algorithm will find w in one iteration with probability $P_{Stern} > 2^{-15.5540}$. If we run the algorithm 2^{18} times then the probability that it will succeed is greater than $1-(1-P_{Stern})^{2^{18}}$, which is more than 99%.

Algorithm 2

```
INPUT: a set of candidates for the matrix $G$
OUTPUT: vector $w$ of weight $n_0 \times d_v$ or "No vector found."

1. for every candidate $\tilde{G}$ for the matrix $G$ do:
   (a) for $i$ from 1 to $2^{18}$ do:
       i. run one iteration of Stern's algorithm on $\tilde{G}$
       ii. if the iteration successfully finds $w$ then:
           A. return $w$
2. return "No vector found."
```

Let us consider Algorithm 2. With probability greater than 99%, Algorithm 2 will find the vector w. On average, the algorithm will have to test $2^{32.5235}$ candidates for the matrix G before finding w. Using the analysis of Stern's algorithm in [8], the cost of one iteration of Stern's algorithm for our parameters is less than $2^{47.9870}$ bit operations. Thus, the expected cost of Algorithm 2 is below $2^{32.5235} \times 2^{18} \times 2^{47.9870} = 2^{98.5105}$.

3.5 Reconstruction of the Private Key

Suppose that Algorithm 2 ran successfully and that the adversary obtained the vector w. The adversary then knows that the secret matrix H has the form as described in Equation (6). In addition, when obtaining the vector w the adversary also learns which candidate for G is the correct one. Thus the adversary knows the correct values of $\tilde{q}_1$, $\tilde{q}_2$, $\tilde{q}_3$, $\tilde{q}_4$, j_2 and j_4 in Equation (5). Thus the adversary knows that the secret matrix Q^T is of the form

$$Q^T = \left[\begin{array}{c|c} x^{k_1}\tilde{q}_1(x) & x^{k_1+l_3}\tilde{q}_3(x) \\ \hline x^{k_1+j_2}\tilde{q}_2(x) & x^{k_1+l_3+j_4}\tilde{q}_4(x) \end{array}\right],$$

for some $k_1, l_3 \in \{0, \dots, p-1\}$, where l_3 is the same l_3 as in Equation (6).

Let

$$\tilde{Q}^T = \left[\begin{array}{c|c} \tilde{q}_1(x) & x^{l_3}\tilde{q}_3(x) \\ \hline x^{j_2}\tilde{q}_2(x) & x^{l_3+j_4}\tilde{q}_4(x) \end{array}\right],$$

and let

$$\tilde{H} = [w^1(x)|x^{p-l_3}w^2(x)].$$

Then

$$\tilde{H}\tilde{Q} = [x^{p-s}]HQ \left[\begin{array}{c|c} x^{k_1} & 0 \\ \hline 0 & x^{k_1} \end{array}\right] = [x^{k_1+p-s}]HQ.$$

Thus $\tilde{H}\tilde{Q}$ is a parity-check matrix for the code generated by the public key G'. Thus matrices $\tilde{H}$ an $\tilde{Q}$ can be used for decrypting ciphertexts. Everything in matrices $\tilde{H}$ an $\tilde{Q}$ is known to the adversary, except for the value of l_3. The value of l_3 can be determined by testing candidates for $\tilde{H}$ an $\tilde{Q}$ against a plaintext-ciphertext pair. The cost of this testing is negligible compared to the cost of Algorithm 2.

References

1. Baldi, M., Barenghi, A., Chiaraluce, F., Pelosi, G., Santini, P.: LEDApkc: Low-dEnsity parity-check coDe-bAsed public-key cryptosystem. Specification revision 1.0 November 30, 2017. Available as a part of the LEDApkc submission package at: https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Round-1-Submissions
2. Baldi, M., Chiaraluce, F.: Cryptanalysis of a new instance of McEliece cryptosystem based on QC-LDPC codes. In: Proc. IEEE ISIT 2007, Nice, France, June 2007, pp. 2591-2595 (2007)
3. Baldi, M., Bodrato, M., Chiaraluce, F.: A new analysis of the McEliece cryptosystem based on QCLDPC codes. In: Ostrovsky, R., De Prisco, R., Visconti, I. (eds.) 6th International Conference on Security and Cryptography for Networks (SCN 2008). LNCS, vol. 5229, pp. 246-262. Springer, Berlin (2008)
4. Fabšič, T., Hromada, V., Stankovski, P., Zajac, P., Guo, Q. and Johansson, T.: A Reaction Attack on the QC-LDPC McEliece Cryptosystem. In: Lange T., Takagi T. (eds) Post-Quantum Cryptography. PQCrypto 2017. Lecture Notes in Computer Science, vol 10346. Springer, Cham (2017)
5. Guo, Q., Johansson, T. and Stankovski, P.: A key recovery attack on MDPC with CCA security using decoding errors. In Advances in CryptologyASIACRYPT 2016: 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I 22 (pp. 789-815). Springer Berlin Heidelberg (2016)
6. Kobara, K. and Imai, H.: Semantically secure McEliece public-key cryptosystems-conversions for McEliece PKC. In: Kim K. (eds) Public Key Cryptography. PKC 2001. Lecture Notes in Computer Science, vol 1992. Springer, Berlin, Heidelberg (2001)
7. Misoczki R., Tillich J-P., Sendrier N., Barreto P.S.L.M.: MDPC-McEliece: new McEliece variants from moderate density parity-check codes. In: IEEE International Symposium on Information Theory (ISIT2013), pp. 2069-2073. Istanbul (2013)
8. Stern, J.: A method for finding codewords of small weight. In: Wolfmann, J., Cohen, G. (eds.) Coding Theory 1988. LNCS, vol. 388, pp. 106-113. Springer, Heidelberg (1989)

Appendix

In figures 3, 4, 5 and 6 we present results from the experiment presented in Algorithm 1 for different values of t.

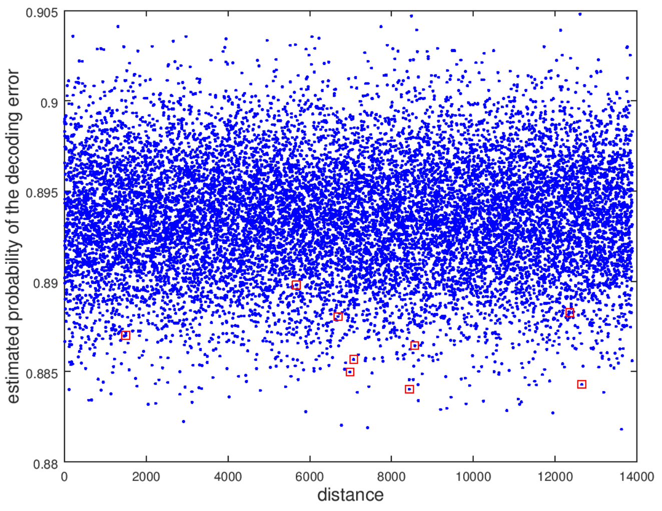

Fig. 3. Values of $\frac{a[d]}{b[d]}$ from the experiment presented in Algorithm 1 with $\mathbf{t} = \mathbf{268}$. The parameter MAX_ENCODABLE_BIT_SIZE_CW_ENCODING was set to 2400. Values associated to distances present in one of the circulant blocks Q_1 and Q_3 are marked by red squares.

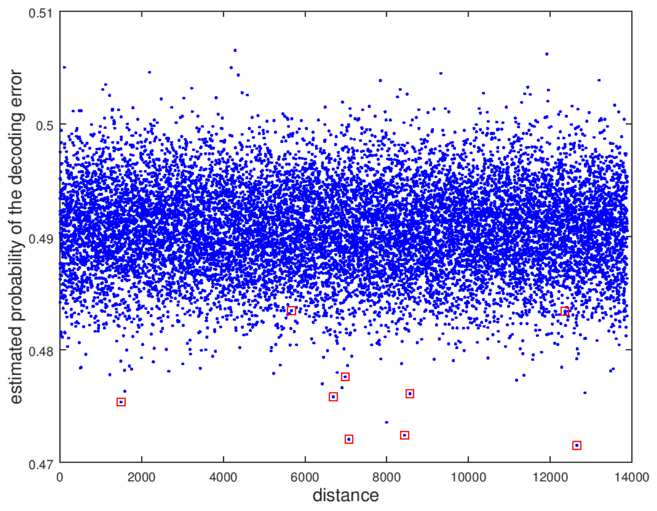

Fig. 4. Values of $\frac{a[d]}{b[d]}$ from the experiment presented in Algorithm 1 with $\mathbf{t} = \mathbf{263}$. The parameter MAX_ENCODABLE_BIT_SIZE_CW_ENCODING was set to 2400. Values associated to distances present in one of the circulant blocks Q_1 and Q_3 are marked by red squares.

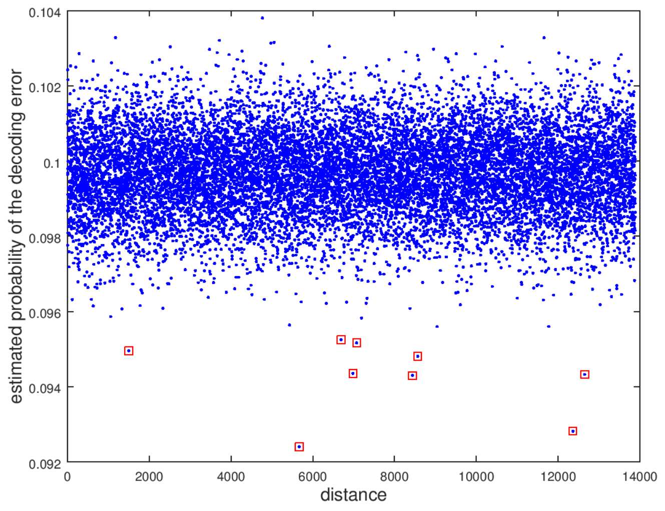

Fig. 5. Values of $\frac{a[d]}{b[d]}$ from the experiment presented in Algorithm 1 with $\mathbf{t = 258}$. The parameter MAX_ENCODABLE_BIT_SIZE_CW_ENCODING was set to 2350. Values associated to distances present in one of the circulant blocks Q_1 and Q_3 are marked by red squares.

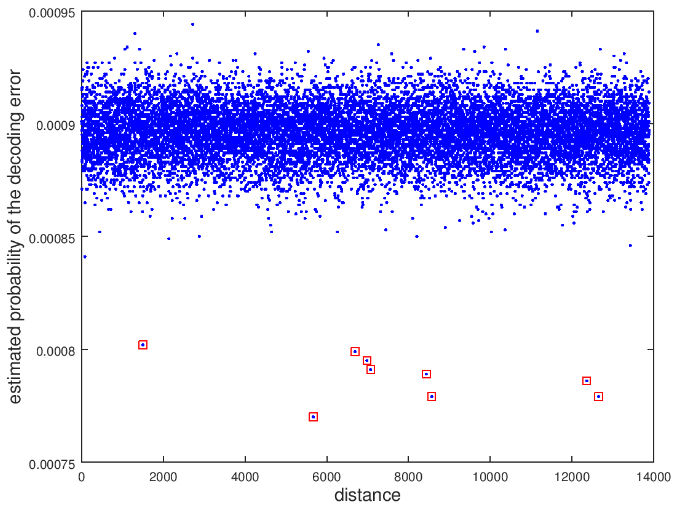

Fig. 6. Values of $\frac{a[d]}{b[d]}$ from the experiment presented in Algorithm 1 with $\mathbf{t = 251}$. The parameter MAX_ENCODABLE_BIT_SIZE_CW_ENCODING was set to 2280. Values associated to distances present in one of the circulant blocks Q_1 and Q_3 are marked by red squares.