Fundamenta Informaticae, Volume 194, Issue 4, Article 2, 2025
DOI: 10.46298/fi.14346
https://fi.episciences.org/





# Privacy for Quantum Annealing. Attack on Spin Reversal Transformations in the Case of Cryptanalysis

**Mateusz Leśniak**
*Department of Cryptology*
*NASK National Research Institute, Kolska Str. 12, Warsaw, Poland*
*mateusz.lesniak@nask.pl*

**Michał Wroński**
*Department of Cryptology*
*NASK National Research Institute, Kolska Str. 12, Warsaw, Poland*
*michal.wronski@nask.pl*

**Abstract.** This paper demonstrates that applying spin reversal transformations, commonly known as a sufficient method for privacy enhancement in problems solved using quantum annealing, does not guarantee privacy for all possible cases. We show how to recover the original problem from the Ising problem obtained using spin reversal transformation when the resulting problem in Ising form represents the algebraic attack on the $E_0$ stream cipher. A small example illustrates how to retrieve the original problem from that transformed by spin reversal transformation. Moreover, we show that our method is efficient also for full-scale problems.

**Keywords:** Privacy enhancement, quantum cloud computing, Ising problem, quantum annealing, secure cloud computing

## 1. Introduction

Quantum optimization is a highly complex process. Despite this, it is gaining considerable popularity [25]. Due to its significant structural requirements, it is usually implemented as a cloud service. The delivered products allow us to solve a wide range of problems. In this paper, we focus on two specific aspects of quantum computation:

- Quantum Annealing, introduced by Tadashi Kadowaki and Hidetoshi Nishimori in [11];

• Quantum Approximate Optimization Algorithm, introduced by Edward Farhi et al. in [7].

As presented in [25, 6], among the application areas of annealing, we can distinguish traffic flow optimization, logistics, vehicle routing problems, finance, and quantum simulation. As interest in quantum optimization grows, so does the need for methods to keep the computations performed private. The classical homomorphic encryption approach [3, 9, 18] does not apply to quantum optimization. In this case, dedicated methods, such as [16], must be used.

In this paper, we focus only on the application method described in [16] and the confidentiality of the data used in the calculations. We attack the scheme shown in [16]. A pen-and-paper example of the attack supports the theoretical description of the attack.

This paper is organized as follows. Section 2 provides an overview of possible privacy-preserving methods, the assumed flow of communication, and a description of the models used. Section 3 describes the attack framework and the attack method's details. Section 4 introduces our attack, while Section 5 demonstrates its practical feasibility using a pen-and-paper example. Finally, Section 6 concludes the paper.

## 2. Background

This section discusses privacy aspects in communication flow and the assumptions made about the adversary. It also presents the basics of optimization models and their transformations.

### 2.1. Quantum cloud services and privacy

Figure 1 considers the communication flow. The client has a specific task that he wants to realize using optimization. In the first step, an optimization problem is formulated for the given task. The following optional step is to encrypt the resulting optimization problem. Assuming ideal communication conditions or having your own quantum computer, this step can be skipped as potentially unnecessary. However, in real-world conditions, failing to do this seriously threatens the privacy of the computations performed. After encryption, the problem is passed to a computer that can perform the indicated optimization. After optimization, a solution is returned. If the problem has been encrypted, it will be necessary to decrypt the solution received further.

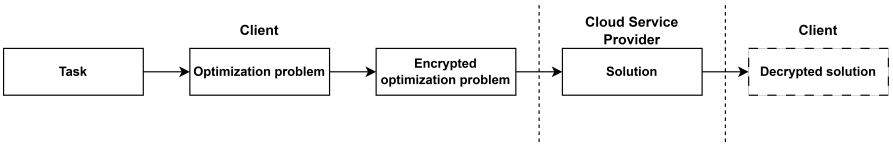


Figure 1. Data flow considered.

Due to the high cost of quantum infrastructure solutions [15], quantum computing is typically implemented using remote access. Such a solution requires the transfer of data to an outsourcer. Due to the inability to verify the cloud service provider's intentions and the lack of control over the data flow on its side, the model discussed below may be assumed.

The adversary is between the client and the quantum cloud service. All communication goes through the adversary, which has the ability to perform any operation on the received problem. This paper assumes that adversaries eavesdrop on intercepted communications and pass them forward.

With the growing number of cloud service providers, the threat is increasing. Service providers are in high demand, and their availability is severely limited. This leads to a situation where a new provider offering competitive services can gain popularity rapidly. The internal structure of cloud services remains a mystery, and it is unknown if there is no eavesdropping between the client and the actual computer. Such a new, untrusted provider may be malicious, and communications between the user and the quantum computer may be intercepted.

In specific cases, such an arrangement is unacceptable. There are few critical applications of quantum optimization where the potential leak of private data involves serious consequences. Transferring the data to an external party is not permissible for such use. In the case of portfolio optimization, such a situation may lead to a financial benefit for the cloud service provider, which could use the optimal solution before passing it to the customer. Another branch of applications for which computational privacy is significant is cryptanalysis. As presented in [10], it may be used in factorizing numbers, solving the discrete logarithm [20, 21, 26, 22, 24] or [5, 23] in algebraic attacks on symmetric ciphers. Computing without privacy in this situation could compromise classified or strategic data.

Just as in the case of classical cloud computing, the solution to the problem is homomorphic encryption [18]; in the case of the discussed issue, the solution may be analogous to specialized privacy-preserving methods. Privacy in quantum computing is a promising area of research. Secure protection algorithms will expand the market for quantum services. As stated in [1], Secure Quantum Computing can be divided into two groups of methods:

- Blind Quantum Computing is a technique that allows outsourcing computations without disclosing the computation's details to the server. This group of protocols is often impractical and requires the customer to have quantum hardware or the participation of several servers.
- Quantum Homomorphic Encryption involves performing calculations on encrypted data. The result of the calculation, after decryption, is the result of the original problem.

This paper focuses on the spin reversal transformation method presented mainly in [16]. The authors propose homomorphic encryption for quantum annealing, which they believe protects the details of quantum annealing instances against a malicious cloud. As stated in the paper, the scheme runs on the Ising model, so it can also be used for privacy preservation in the Quantum Approximate Optimization Algorithm.

## 2.2. A description of the models used

As mentioned earlier, using the Quantum Approximate Optimization Algorithm or Quantum Annealing requires presenting the problem in a specific form. One such form is the Ising model, which has existed since 1920. Ernst Ising and Wilhelm Lenz introduced it as a description of magnetic materials. Despite its original application, the model has also been applied in combinatorial optimization. As mentioned in [19], each Ising problem can be viewed as a minimizing expression presented as

Equation (1)

$$f(s) = \sum_{i=0}^{n-1} h_i s_i + \sum_{i,j=0}^{n-1} J_{i,j} s_i s_j. \tag{1}$$

Vector $s$ is called the state, and each variable $s_i \in \{-1, 1\}$ is called a spin. In practice, the following are used to characterize the problem:

- a vector of biases:

$$h = \begin{bmatrix} h_0 & h_1 & \cdots & h_{n-1} \end{bmatrix}^T; \tag{2}$$

- a matrix describing connections between variables:

$$J = \begin{bmatrix} J_{0,0} & J_{0,1} & \cdots & J_{0,n-1} \\ J_{1,0} & J_{1,1} & \cdots & J_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ J_{n-1,0} & J_{n-1,1} & \cdots & J_{n-1,n-1} \end{bmatrix}. \tag{3}$$

In this case, communication with the quantum cloud requires sending vector of biases $h$ and a connection matrix $J$, presented as Equations (2) and (3) respectively.

An alternative model is the QUBO model, which uses binary variables $x \in \{0, 1\}$. Quadratic unconstrained binary optimization can also be viewed as minimizing the specific expression shown as Equation (4)

$$f(x) = \sum_{i=0}^{n-1} Q_{i,i} x_i + \sum_{i,j=0}^{n} Q_{i,j} x_i x_j, \tag{4}$$

where vector $x$ contains $n$ binary variables.

As presented in [14], there exist Ising formulations of many NP problems, such as graph partitioning [8], the knapsack problem [12], graph coloring [12], or the traveling salesman [12] problem. However, it is easier for some tasks to formulate the problem in QUBO form. Such tasks include algebraic attacks on symmetric ciphers, as described in [5, 4, 23]. Applying privacy-preserving methods for such problems may require transitioning between the QUBO and Ising models and vice versa.

There is a simple transformation between the Ising model and the QUBO problem. Its main idea is to perform the following substitutions to change spin variables into binary variables. When moving from the Ising model to the QUBO:

$$x_i = \frac{1}{2} \cdot (s_i + 1).$$

When moving from the QUBO to the Ising model:

$$s_i = 2x_i - 1.$$

As described in [1], coefficients of each matrix can be determined using matrix coefficients for the corresponding problem. When moving from the Ising model to the QUBO problem, the connection matrix is determined as follows:

$$\begin{aligned} Q_{i,j} &= 4J_{i,j}, \\ Q_{i,i} &= 2\big(h_i - \textstyle\sum_j J_{i,j} - \sum_j J_{j,i}\big). \end{aligned} \tag{5}$$

At the transition in the opposite direction, the bias vector is determined as follows:

$$h_i = \frac{Q_{i,i}}{2} + \frac{\sum_j Q_{i,j} + \sum_j Q_{j,i}}{4}. \tag{6}$$

The connection matrix is determined as follows:

$$J_{i,j} = \frac{Q_{i,j}}{4}. \tag{7}$$

In the rest of the paper, we skip transitions between models, considering this operation trivial.

# 3. Spin Reversal Transformation in detail

This section briefly describes the application of spin reversal transformation to preserve privacy. This idea is mainly presented in [16] and partly in [1]. The method utilizes a random sequence to reverse a given sign in an Ising problem instance. According to the authors, an adversary, having intercepted a concealed problem, cannot reconstruct the original problem without knowing the original key.

## 3.1. Description of the algorithm

The encryption scheme described in [16] is based on spin reversal transformation, also called a gauge transformation [17]. The transformation uses a binary string $x$ to change the signs of the selected coefficients. After applying the mentioned transformation, Equation (1) is transformed as follows:

$$f^*(s^*) = \sum_{i=0}^{n-1} (-1)^{x_i} h_i s_i^* + \sum_{i,j=0}^{n-1} (-1)^{x_i + x_j} J_{i,j} s_i^* s_j^*. \tag{8}$$

The same sequence $x$ must be used to determine the original solution. The corresponding solution can be determined according to Equation 9:

$$s_i = (-1)^{x_i} s_i^*. \tag{9}$$

It is important to note that the minimal energy of the instance does not change. As shown in [16] and [17], the solution to the concealed problem, when uncovered, is the solution to the original problem.

The described transformation can be outlined as a simple scheme, as in [16]:

1. The client generates a secret key $x = (x_0, x_1, \ldots, x_{n-1})$;
2. The client computes $h^*$ and matrix $J^*$:

$$h_i^* = (-1)^{x_i} h_i, \tag{10}$$

$$J_{i,j}^* = (-1)^{x_i + x_j} J_{i,j}, \tag{11}$$

3. The client sends the concealed problem to the quantum cloud service and receives the solution $s^*$;
4. The client retrieves the solution using Equation (9).

# 4. Details of the proposed attack

This paper argues that the privacy-enhancing mechanism presented in [16] fails in many instances. As a counterexample, we demonstrate that the method fails when the technique of transforming the stream cipher cryptanalytic problem is known. Specifically, we show that, without prior knowledge of the random sequence used to conceal the given Ising problem for cryptanalysis of the $E_0$ cipher, one can easily retrieve the original problem using the intercepted data. Furthermore, by knowing the solution to the concealed problem, one can also deduce the solution to the original problem. It also allows the recovery of the concealment key and the data on which the client wants confidentiality (the key hidden in the solution to the problem sent, ciphertexts, or keystream). With the concealment key, the adversary can recover communications encrypted with this key. The design of our attack involves three phases:

1. **Parameterization (optional)**: Based on knowledge of the type of optimization task and access to the oracle $\phi$, a parameterized matrix is constructed. It can be implemented before the attack and only once for a given optimization task. Subsequent attacks use the predetermined matrix.
2. **System setup**: Using the parameterized matrix and the intercepted encrypted optimization problem $h_i^*, J_{i,j}^*$, a system of linear equations is created.
3. **Solving**: The obtained system of equations is solved, determining the keystream used to create the optimization task and the concealment key.

## 4.1. The problem for the $E_0$ cipher

We present an attack on the mentioned scheme using the QUBO problem corresponding to the cryptanalysis of the $E_0$ cipher, presented in [13]. As the authors point out, the reduction shown in that paper in the next few years can be practically embedded in a commercially available quantum annealer. This opens up the possibility of application to real cryptanalysis, for which it is essential to maintain the privacy of transmitted data and recovered keys.

Our proposed attack can also be applied to other highly structured optimization problems. The following example was chosen for its potential practicality and its scalability.

Below is a brief overview of the construction of the $E_0$ cipher. A full description of the cipher can be found in [2]. The cipher is built from three elements:

- Four shift registers with linear feedback, specified by the following primitive polynomials $f_i(x)$:

$$\begin{aligned} L_1: &\quad f_1(x) = x^{25} \oplus x^{20} \oplus x^{12} \oplus x^8 \oplus 1, \\ L_2: &\quad f_2(x) = x^{31} \oplus x^{24} \oplus x^{16} \oplus x^{12} \oplus 1, \\ L_3: &\quad f_3(x) = x^{33} \oplus x^{28} \oplus x^{24} \oplus x^4 \oplus 1, \\ L_4: &\quad f_4(x) = x^{39} \oplus x^{36} \oplus x^{28} \oplus x^4 \oplus 1. \end{aligned}$$

- Summation Combiner Logic, computing the two-bit value $s_{t+1}$:

$$F_1 : s_{t+1} = \left\lfloor \frac{\sum_{i=1}^{4} x_i + c_t}{2} \right\rfloor.$$

- Blend Register, calculating the two-bit value of $c_{t+1}$ using the bijections described in the cipher specification:

$$F_2 : c_{t+1} = s_{t+1} \oplus T_1[c_t] \oplus T_2[c_{t-1}],$$

where $T_i$ are linear mappings:

$$\begin{aligned} T_1 : &\quad (x_1, x_0) \to (x_1, x_0), \\ T_2 : &\quad (x_1, x_0) \to (x_0, x_1 \oplus x_0). \end{aligned}$$

- Each keystream bit is determined by the value from shift registers with linear feedback, $x_0$, $x_1$, $x_2$, $x_3$, and one bit from the Blend Register:

$$z_i = x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus c_t^0.$$

Each bit of the keystream is described by eight equations:

$$\begin{cases} f_0 : z_t = l_{t+1} \oplus m_{t+7} \oplus n_{t+1} \oplus o_{t+7} \oplus c_t^0, \\ f_1 : l_{t+25} = l_t \oplus l_{t+5} \oplus l_{t+13} \oplus l_{t+17}, \\ f_2 : m_{t+31} = m_t \oplus m_{t+7} \oplus m_{t+15} \oplus m_{t+19}, \\ f_3 : n_{t+33} = n_t \oplus n_{t+5} \oplus n_{t+9} \oplus n_{t+29}, \\ f_4 : o_{t+39} = o_t \oplus o_{t+3} \oplus o_{t+11} \oplus o_{t+35}, \\ f_5 : c_{t+1}^1 = s_{t+1}^1 \oplus c_t^1 \oplus c_{t-1}^0, \\ f_6 : c_{t+1}^0 = s_{t+1}^0 \oplus c_t^0 \oplus c_{t-1}^1 \oplus c_{t-1}^0, \\ f_7 : 4s_{t+1}^1 + 2s_{t+1}^0 + \beta = l_{t+1} + m_{t+7} + n_{t+1} + o_{t+7} + 2c_t^1 + c_t^0. \end{cases} \tag{12}$$

## 4.2. From algebraic attack to quantum optimization

This section briefly describes the transformation of the algebraic attack to the QUBO optimization problem. The main idea was presented in [5] and adapted to $E_0$ in [13]. In the first step, a system of equations is generated. To recover the initial internal state of the cipher, we need 128 bits of the keystream. As mentioned, each stream bit is described by eight equations. In total, the system describing the cipher will have 1024 equations. Below, we show how to perform transformations of the resulting system in a few steps:

1. Equations $f_i, i = \overline{0,6}$ are transformed into equations with binary variables and integer coefficients. Equation $f_7$ does not require such a transformation. From the construction of the cipher, it is already in this form:

$$f_i' \equiv 0 (\text{mod } 2) \to f_i - 2k_i = 0.$$

Each integer variable $k_i$ is bounded, $k_i \leq \lfloor \frac{f_i^{\max}}{2} \rfloor$, where $f_i^{\max}$ is the maximum value of the selected polynomial.

2. In the standard transformation, the equations must be linearized in the next step. However, this step is skipped in this paper; the $E_0$ design induces no nonlinear equations.

3. Then variables $k_i$ are replaced with binary variables. Each $k_i$ requires $bl(f_i^{\max})$ new binary variables, where $bl(x)$ denotes the bit-length of $x$.
4. In the final step, polynomial $F'_{\text{Pen}}$ is determined according to Equation (13):

$$F'_{\text{Pen}} = \sum_{i=0}^{m-1} (f'_i)^2, \tag{13}$$

The standard components with a penalty are omitted because they are zero. Finally, the constant present in the polynomial is subtracted from the resulting polynomial.

The final matrix obtained has a size of $N = 2728$ variables and $20598$ non-zero coefficients, which is $0.55\%$ of all matrix elements.

### 4.3. How to identify the problem and parameterize it?

Quantum service providers require data to identify the user. Among such data, we can distinguish:

- first and last name;
- email address;
- company name, job title, field of study;
- the purpose of computer access.

Using them, an untrusted provider can identify the client and the research area in which the client works. Identifying the area of study will allow the data obtained to be matched with one of the known problems. Problems vary significantly in size, density, and coefficient range. We assume that resources cannot freely change the connectivity (density) or size of the problem.

The lowest layer and critical element of the attack is the parameterization of the matrix of the selected optimization problem. The parameterization details will depend on the type of optimization task. However, the general principle remains the same. Below is an idea of how it can be realized for an algebraic attack on an $E_0$ cipher.

As described earlier, the selected optimization problem depends on the given keystream. To parameterize its matrix, we need to identify the coefficients of the matrix that depend on the $z_i$. As shown in Equation (12), only one of the equations depends on the bits of the keystream. We highlight two methods: the equation analysis dependent on $z_i$ and the algorithmic approach based on the methods of construction of the optimization problem, where an oracle $\phi$ is created.

The first method requires analyzing the equations. According to the method of transforming the algebraic attack described earlier, we focus on Equation (13). For each bit of the keystream, there will be a component in the final polynomial shown in:

$$z_t + l_{t+1} + m_{t+7} + n_{t+1} + o_{t+7} + c_t^0 - 2K = 0 \Big/^2.$$

After squaring, we get six coefficients depending on the keystream bits: $2z_t \cdot l_{t+1}, 2z_t \cdot m_{t+7}, 2z_t \cdot n_{t+1}, 2z_t \cdot o_{t+7}, 2z_t \cdot c_t^0, -4z_t \cdot K$. Note that $K$ is an integer variable and should be replaced with a binary variable. The coefficients resulting from the squaring of other equations are constant. With the above knowledge and using Equation (6), equations dependent on the keystream occurring in the vector of biases can be computed.

The alternative method does not require direct analysis of equations. It is more generic and can be used universally for any problem for which the construction of the optimization problem is known. The coefficients of the matrix depend linearly on secret information. This is the situation for the $E_0$ cipher and most stream ciphers. Assume that an oracle $\phi$ is given which, for any keystream, will return an Ising model, as in Equation (15). The oracle can be constructed based on known publications on selected optimization problems, similar to the earlier description in Section 4.2. The chosen method requires $|z| + 1$ queries to create a parameterized matrix for the keystream of length $|z|$.

To determine the coefficients depending on a specific bit of the keystream, we use all streams with a Hamming weight of 1. Such a stream can be denoted as $s_i$ and is presented as Equation (14):

$$s_i = \{\underbrace{0, \ldots, 0}_{i-1}, \underset{i}{1}, \underbrace{0, \ldots, 0}_{n-i}\}. \tag{14}$$

We denote the Ising models corresponding to the streams $s_i$ as $h_i$ and $J_i$. They are determined using the oracle $\phi$, as shown in Equation (15):

$$h_i, J_i \leftarrow \phi(s_i). \tag{15}$$

Additionally, we denote $h_\infty$ and $J_\infty$ as the Ising model for a stream with Hamming weight equal to zero. Then, the parameterized problem is computed as follows:

$$\begin{aligned} h^P &= h_\infty + \textstyle\sum_{i=0}^{n}(h_i - h_\infty) \cdot z_i, \\ J^P &= J_\infty + \textstyle\sum_{i=0}^{n}(J_i - J_\infty) \cdot z_i, \end{aligned} \tag{16}$$

where $z_i$ is a variable.
The idea of this parameterization method is illustrated in the example in Section 5.1.

## 4.4. Executing the attack, the most straightforward phase

An attack is performed using a parameterized problem. A vector of biases is sufficient to perform the attack.

First, a system of linear equations is constructed. The encrypted vector of biases $h^*$ is juxtaposed with a parameterized one $h^P$ to form a system of equations:

$$h^* = h^P = \begin{bmatrix} h_0^* \\ h_1^* \\ \vdots \\ h_N^* \end{bmatrix} = \begin{bmatrix} h_0^P \\ h_1^P \\ \vdots \\ h_N^P \end{bmatrix}.$$

As the design of the parameterized matrix shows, each element of $h^P$ can be presented as a sum of rational number $\texttt{const}_i$ and a linear combination of selected bits of the keystream. We denote the set of indices of the relevant bits for element $i$ as $\mathcal{A}_i$. The equations formed in the proposed way can be represented by Equation (17):

$$h_i^* = \sum_{j \in \mathcal{A}_i} z_j + \texttt{const}_i, \tag{17}$$

where $\texttt{const}_i \in \mathbb{Q}$.

The next step is to analyze each resulting equation in the correct order. According to how the problem is constructed, there will be more equations than unknowns in the created matrix. From among all the equations, we choose some set of sufficient size and reduce the equations to the form presented in Equation (18):

$$z_k + b_i = h_i^*, \tag{18}$$

where $b_i \in \mathbb{Q}$ is sum of $\texttt{const}_i$ and known keystream bits. As $k$ we denotes the index of the keystream variable occurring in this equation.

If the transformation of all the equations is impossible, we rearrange the system to an upper triangular form. Then, an equation analysis is performed, starting with the last equation. In the next steps, successive equations will be reduced, considering the previous solutions and performing an analysis.

As can be seen from the definition of the optimization problem under consideration, the variable $z_k$ is a binary variable, $z_k \in \{0, 1\}$. Therefore, the relation (19) should be satisfied:

$$h_i^* - b_i \in \{0, 1\}. \tag{19}$$

Due to the use of encryption, two situations can occur: the condition will be met or not. If the above relation is satisfied, then:

- the relevant coefficient has not been concealed;
- the designated bit of the concealment key is $x_i = 0$;
- the designated bit of the keystream is $z_k = h_i^* - b_i$.

Otherwise, the given coefficient is concealed and:

- the designated bit of the concealment key is $x_i = 1$;
- the designated bit of the keystream is $z_k = -h_i^* - b_i$.

In most cases, solving the equation above gives the bit of the keystream $z_k$ with a probability of 1. If one finds that $z_k \in \{0, 1\}$ regardless of whether a coefficient is concealed, one can check if other equations where $z_k$ appears are correct.

The above procedure allows the recovery of the used keystream. The remaining bits of the concealment key can be determined by comparing the remaining coefficients. If the coefficients are opposite, the given coefficient has been concealed, and the corresponding concealment key bit is 1. For every bit of the keystream, retrieving the correct value of the bit requires only solving an affine equation, and the entire attack is swift. In addition, the attack allows the recovery of the concealment key. With the use of this key:

- any further problem encrypted with the same key can be exposed;
- the solution to the problem sent may be unveiled, including knowledge of the secret key that the client wanted to recover.

## 5. A practical example of the proposed attack

To illustrate the correctness of the attack, an attack performed in practice is presented. A scaled-down $E_0$ cipher was used to show the step-by-step operation of the attack. In addition, the execution of the attack on the full version is discussed.

## 5.1. Example parameterization

In this section, we present a pen-and-paper example of parameterization. We are given an oracle $\phi$ that returns bias vectors for given streams generated by a hypothetical stream cipher. Here, we focus on supporting the idea of parameterization with an example, so we omit the details of the hypothetical cipher. Assume that we are looking for a parameterized matrix corresponding to an algebraic attack using a 3-bit keystream $(z_0, z_1, z_2)$. We now follow the description presented earlier:

1. We determine the bias vector for a stream with a Hamming weight of 0:

$$h_\infty = \phi(0,0,0) = \begin{bmatrix} 2 \\ 4 \\ 1 \end{bmatrix}.$$

2. We determine the bias vector for all streams with a Hamming weight of 1:

$$h_0 = \phi(1,0,0) = \begin{bmatrix} 3 \\ 5 \\ 2 \end{bmatrix}, \quad h_1 = \phi(0,1,0) = \begin{bmatrix} 2 \\ 5 \\ 1 \end{bmatrix}, \quad h_2 = \phi(0,0,1) = \begin{bmatrix} 2 \\ 4 \\ 7 \end{bmatrix}.$$

3. We determine the differences $h_i - h_\infty$, where $i \in \{0,1,2\}$:

$$h_0 - h_\infty = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \quad h_1 - h_\infty = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad h_2 - h_\infty = \begin{bmatrix} 0 \\ 0 \\ 6 \end{bmatrix}.$$

4. We determine the parameterized vector as:

$$h^P = h_\infty + (h_0 - h_\infty) \cdot \mathtt{z_0} + (h_1 - h_\infty) \cdot \mathtt{z_1} + (h_2 - h_\infty) \cdot \mathtt{z_2},$$

so:

$$h^P = \begin{bmatrix} 2 \\ 4 \\ 1 \end{bmatrix} + \begin{bmatrix} \mathtt{z_0} \\ \mathtt{z_0} \\ \mathtt{z_0} \end{bmatrix} + \begin{bmatrix} 0 \\ \mathtt{z_1} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 6\mathtt{z_2} \end{bmatrix} = \begin{bmatrix} 2 + \mathtt{z_0} \\ 4 + \mathtt{z_0} + \mathtt{z_1} \\ 1 + \mathtt{z_0} + 6\mathtt{z_2} \end{bmatrix}.$$

The searched parameterized matrix of the assumed algebraic attack was thus determined using four oracle queries.

## 5.2. An illustrative attack on a scaled-down instance

An instance of the cipher using LFSRs described by the following primitive polynomials was used for the attack:

- $L_1 : f_1(x) = x^3 + x + 1$,
- $L_2 : f_2(x) = x^3 + x + 1$,
- $L_3 : f_3(x) = x^3 + x + 1$,
- $L_4 : f_4(x) = x^3 + x + 1$.

To simplify the generated problem as much as possible, the registers of the scaled cipher should be of equal length. The set of primitive polynomials of degree 3 is limited. For this reason, the selected polynomials are equal. The impact of such a solution on the cipher's security is not the subject of this paper. Such a configuration allows a pictorial representation of the proposed attack.

For the selected cipher instance, the following keystream was used:

$$z = (0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0). \tag{20}$$

For the given sequence, 96 equations were generated. Based on these equations, a QUBO problem with 240 variables was determined. The problem was transformed into an Ising model and encrypted using the idea presented in [16], as described in Section 3. Due to the size of the matrix, we cannot include the entire key here. Below, as Equation (21), we present selected key bits corresponding to the coefficients of the vector at the positions analyzed:

$$x = (\underset{121}{1}, \underset{131}{1}, \underset{141}{1}, \underset{151}{0}, \underset{161}{1}, \underset{171}{1}, \underset{181}{0}, \underset{191}{1}, \underset{201}{1}, \underset{211}{0}, \underset{221}{0}, \underset{231}{1}). \tag{21}$$

Values under consecutive bits indicate the position number of the specified key bit.

An oracle was built based on [13]. Using it, a parameterized problem was generated according to the idea in Section 4.3. To obtain the entire parameterized bias vector, 13 queries were performed. The equations presented as Equation (22) are selected to recover the keystream from this vector. As described in Section 4.4, such a set of equations was chosen so that it would be possible to analyze them one by one and determine the entire keystream. Out of 240 equations in a parameterized matrix, 12 were selected for further analysis:

$$\begin{cases} f_{121} : -z_0 - 1 = 1, \\ f_{131} : -z_1 - 1 = 1, \\ f_{141} : -z_2 - 1 = 2, \\ f_{151} : -z_3 - 1 = -1, \\ f_{161} : -z_4 - 1 = 2, \\ f_{171} : -z_5 - 1 = 2, \\ f_{181} : -z_6 - 1 = -2, \\ f_{191} : -z_7 - 1 = 2, \\ f_{201} : -z_8 - 1 = 1, \\ f_{211} : -z_9 - 1 = -1, \\ f_{221} : -z_{10} - 1 = -2, \\ f_{231} : -z_{11} - 1 = 1. \end{cases} \tag{22}$$

The equation analysis is shown in Table 1. Based on it, the coefficients with changed signs were determined. Finally, based on the above analysis, the keystream was determined.

The relevant part of the concealment key was determined using the remaining elements of the vector. The recovered key, the result of the developed script implementing the attack, is shown in

Table 1. Analysis of selected equations. Determination of part of the key bits.

| $i$ | Equation | $z_i \in \{0, 1\}$ | $x_i$ |
|---|---|---|---|
| 121 | $-z_0 - 1 = 1 \rightarrow z_0 = -2$ | ✗ | 1 |
| 131 | $-z_1 - 1 = 1 \rightarrow z_1 = -2$ | ✗ | 1 |
| 141 | $-z_2 - 1 = 2 \rightarrow z_2 = -3$ | ✗ | 1 |
| 151 | $-z_3 - 1 = -1 \rightarrow z_3 = 0$ | ✓ | 0 |
| 161 | $-z_4 - 1 = 2 \rightarrow z_4 = -3$ | ✗ | 1 |
| 171 | $-z_5 - 1 = 2 \rightarrow z_5 = -3$ | ✗ | 1 |
| 181 | $-z_6 - 1 = -2 \rightarrow z_6 = 1$ | ✓ | 0 |
| 191 | $-z_7 - 1 = 2 \rightarrow z_7 = -3$ | ✗ | 1 |
| 201 | $-z_8 - 1 = 1 \rightarrow z_8 = -2$ | ✗ | 1 |
| 211 | $-z_9 - 1 = -1 \rightarrow z_9 = 0$ | ✓ | 0 |
| 221 | $-z_{10} - 1 = -2 \rightarrow z_{10} = 1$ | ✓ | 0 |
| 231 | $-z_{11} - 1 = 1 \rightarrow z_{11} = -2$ | ✗ | 1 |

```
Retrieved key:
[1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1,
1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 'x', 0, 'x', 0, 'x', 0, 'x', 1, 'x', 1, 'x', 1, 'x', 1, 'x', 0, 'x', 0, 'x', 1,
'x', 1, 'x', 'x', 'x', 'x', 'x', 'x', 'x', 'x', 'x', 'x', 'x', 'x', 'x', 'x', 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0,
1, 1, 1, 1, 1, 1, 0, 'x', 1, 'x', 1, 1, 0, 1, 0, 1, 'x', 'x', 0, 0, 0, 1, 0, 0, 0, 1, 'x', 'x', 1, 0, 0, 1, 1, 1, 0, 1, 'x',
'x', 0, 1, 0, 0, 1, 1, 0, 1, 'x', 'x', 0, 0, 0, 1, 0, 0, 1, 0, 'x', 'x', 0, 1, 0, 1, 1, 1, 'x', 'x', 1, 1, 0, 0, 0, 1, 0,
1, 'x', 'x', 1, 0, 1, 1, 1, 0, 0, 0, 'x', 'x', 0, 0, 0, 1, 1, 1, 1, 1, 'x', 'x', 1, 0, 1, 0, 0, 1, 0, 1, 'x', 'x', 0, 0, 1, 0,
1, 0, 0, 0, 'x', 'x', 1, 0, 1, 1, 0, 0, 1, 0, 'x', 'x', 1, 0]
```

Figure 2. Key recovered as a result of the experiment.

Figure 2. Missing bits, denoted as $x$ in Figure 2, of the key do not affect the ciphertext. It is impossible to change the sign of a coefficient equal to 0.

We can verify the correctness in two ways:

- performing a comparison of the sequence shown in Equation (21) with the last column of Table 1;
- using the recovered keystream, the full bias vector is determined. Using the recovered key, the concealed vector is decrypted. Then, a comparison of both vectors is performed.

Both of these methods confirm the correctness of the attack performed.

## 5.3. Attack on the full version of the $E_0$ cipher

An attack on an encrypted problem corresponding to the cryptanalysis of the full $E_0$ cipher can be performed similarly. Using a script in the SageMath environment:

1. A problem corresponding to an algebraic attack on the $E_0$ cipher is generated for the selected keystream. The problem has 2728 variables, and the used keystream has 128 bits.

2. The resulting optimization problem is encrypted using Spin Reversal Transformation with a randomly generated key.
3. A parameterized matrix is determined. It requires 129 oracle calls. The time needed to generate the parameterized matrix can be estimated at about 13.5 hours.
4. Based on the determined matrix, a system of 128 linear equations is arranged.
5. In the last step, the designated equations are analyzed one by one, and the key used and the stream for which the problem was generated is determined.
6. Finally, the results obtained are verified. The recovered keystream is compared with the original one. The matrix is decrypted and compared with the problem generated for the recovered keystream.

As in the rescaled example, the longest step is the parameterization of the matrix. However, the duration of the attack still allows it to be performed in practice.

It should be noted that the script used is not optimized in any way, and the experiment only illustrates the disparity between the time of the actual phase of the attack and the time of the pre-computations.

## 6. Summary and future work

This paper shows a practical attack on a proposed scheme to ensure the privacy of problems sent to quantum computing clouds. In addition, a practical attack on a miniature version of the cipher was demonstrated to show the correctness of the attack. It should be noted that the work does not address the correctness of the scheme under attack. The proposed attack was prepared with assumptions in the original work and additional ones resulting from the functioning of available services.

Further work should look for a secure way to protect cloud computing based on the Ising model. As an alternative to using an untrusted provider, methods can be developed that allow local problem solving using private infrastructure.

## References

[1] Ayanzadeh R, Mousavi A, Alavisamani N, Qureshi M. Enigma: Privacy-Preserving Execution of QAOA on Untrusted Quantum Computers, 2023. 2311.13546.

[2] Bluetooth Special Interest Group. Bluetooth Core Specification, 2021. Rev. 5.3.

[3] Brakerski Z, Gentry C, Vaikuntanathan V. (Leveled) Fully Homomorphic Encryption without Bootstrapping. *ACM Trans. Comput. Theory*, 2014. **6**(3). doi:10.1145/2633600. URL https://doi.org/10.1145/2633600.

[4] Burek E, Wroński M. Quantum Annealing and Algebraic Attack on Speck Cipher. In: International Conference on Computational Science. Springer, 2022 pp. 143–149.

[5] Burek E, Wroński M, Mańk K, Misztal M. Algebraic attacks on block ciphers using quantum annealing. *IEEE Transactions on Emerging Topics in Computing*, 2022. **10**(2):678–689.

[6] D-Wave Systems. Hundreds of Quantum Applications, 2023. URL https://www.dwavesys.com/learn/featured-applications/.

[7] Farhi E, Goldstone J, Gutmann S. A Quantum Approximate Optimization Algorithm, 2014. 1411.4028, URL https://arxiv.org/abs/1411.4028.

[8] Fu Y, Anderson PW. Application of statistical mechanics to NP-complete problems in combinatorial optimisation. *Journal of Physics A: Mathematical and General*, 1986. **19**(9):1605. doi:10.1088/0305-4470/19/9/033. URL https://dx.doi.org/10.1088/0305-4470/19/9/033.

[9] Gentry C, Sahai A, Waters B. Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based. In: Canetti R, Garay JA (eds.), Advances in Cryptology – CRYPTO 2013. Springer Berlin Heidelberg, Berlin, Heidelberg. ISBN 978-3-642-40041-4, 2013 pp. 75–92.

[10] Jiang S, Britt KA, McCaskey AJ, Humble TS, Kais S. Quantum Annealing for Prime Factorization. *Scientific Reports*, 2018. **8**(1):17667. doi:10.1038/s41598-018-36058-z. URL https://doi.org/10.1038/s41598-018-36058-z.

[11] Kadowaki T, Nishimori H. Quantum annealing in the transverse Ising model. *Phys. Rev. E*, 1998. **58**:5355–5363. doi:10.1103/PhysRevE.58.5355. URL https://link.aps.org/doi/10.1103/PhysRevE.58.5355.

[12] Karp, Richard M. Reducibility among Combinatorial Problems, pp. 85–103. Springer US, Boston, MA. ISBN 978-1-4684-2001-2, 1972. doi:10.1007/978-1-4684-2001-2_9. URL https://doi.org/10.1007/978-1-4684-2001-2_9.

[13] Leśniak M, Burek E, Wroński M. Unsafe Mechanisms of Bluetooth, $E_0$ Stream Cipher Cryptanalysis with Quantum Annealing. In: Franco L, de Mulatier C, Paszynski M, Krzhizhanovskaya VV, Dongarra JJ, Sloot PMA (eds.), Computational Science – ICCS 2024. Springer Nature Switzerland, Cham. ISBN 978-3-031-63778-0, 2024 pp. 389–404.

[14] Lucas A. Ising formulations of many NP problems. *Frontiers in Physics*, 2014. **2**. doi:10.3389/fphy.2014.00005. URL https://www.frontiersin.org/articles/10.3389/fphy.2014.00005.

[15] Memon QA, Al Ahmad M, Pecht M. Quantum Computing: Navigating the Future of Computation, Challenges, and Technological Breakthroughs. *Quantum Reports*, 2024. **6**(4):627–663. doi:10.3390/quantum6040039. URL https://www.mdpi.com/2624-960X/6/4/39.

[16] O'Malley D, Golden JK. Homomorphic Encryption for Quantum Annealing with Spin Reversal Transformations. In: 2020 IEEE High Performance Extreme Computing Conference (HPEC). 2020 pp. 1–6. doi:10.1109/HPEC43674.2020.9286176.

[17] Pelofske E, Hahn G, Djidjev H. Optimizing the Spin Reversal Transform on the D-Wave 2000Q. In: 2019 IEEE International Conference on Rebooting Computing (ICRC). 2019 pp. 1–8. doi:10.1109/ICRC.2019.8914719.

[18] Ronald L Rivest, Michael L Dertouzos. On Data Banks and Privacy Homomorphism. 1978 URL https://api.semanticscholar.org/CorpusID:6905087.

[19] Vertogen G, de Vries, A S. The Ising problem. *Communications in Mathematical Physics*, 1973. **29**(2):131–162.

[20] Wroński M. Index calculus method for solving elliptic curve discrete logarithm problem using quantum annealing. In: International Conference on Computational Science. Springer, 2021 pp. 149–155.

[21] Wroński M. Practical Solving of Discrete Logarithm Problem over Prime Fields Using Quantum Annealing. In: Groen D, de Mulatier C, Paszynski M, Krzhizhanovskaya VV, Dongarra JJ, Sloot PMA (eds.), Computational Science – ICCS 2022. Springer International Publishing, Cham. ISBN 978-3-031-08760-8, 2022 pp. 93–106.

[22] Wroński M, Burek E, Dzierzkowski Ł, Żołnierczyk O. Transformation of Elliptic Curve Discrete Logarithm Problem to QUBO Using Direct Method in Quantum Annealing Applications. *Journal of Telecommunications and Information Technology*, 2024. **95**(1):75–82. doi:10.26636/jtit.2024.1.1463. URL https://www.jtit.pl/jtit/article/view/1463.

[23] Wroński M, Burek E, Leśniak M. (In)Security of Stream Ciphers Against Quantum Annealing Attacks on the Example of the Grain 128 and Grain 128a Ciphers. *IEEE Transactions on Emerging Topics in Computing*, 2024. pp. 1–14. doi:10.1109/TETC.2024.3474856. URL https://doi.org/10.1109/TETC.2024.3474856.

[24] Wroński M, Dzierzkowski Ł. Base of exponent representation matters-more efficient reduction of discrete logarithm problem and elliptic curve discrete logarithm problem to the QUBO problem. *Quantum Information and Computation*, 2024. **24**(7&8):0541–0564.

[25] Yarkoni S, Raponi E, Bäck T, Schmitt S. Quantum annealing for industry applications: introduction and review. *Reports on Progress in Physics*, 2022. **85**(10):104001. doi:10.1088/1361-6633/ac8c54. URL https://dx.doi.org/10.1088/1361-6633/ac8c54.

[26] Żołnierczyk O, Wroński M. Searching B-Smooth Numbers Using Quantum Annealing: Applications to Factorization and Discrete Logarithm Problem. In: International Conference on Computational Science. Springer, 2023 pp. 3–17.